UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Darja Eržen

# Zasnova priporočilnega sistema za podporo izbora študijskih predmetov

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Blaž Zupan

Ljubljana, 2012

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

*Tržaška 25*
*1001 Ljubljana, Slovenija*
*telefon: 01 476 84 11*
*01 476 83 87*
*faks: 01 426 46 47*
*01 476 87 11*
*www.fri.uni-lj.si*
*e-mail: dekanat@fri.uni-lj.si*

Št. naloge: 01814/2012
Datum: 15.03.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **DARJA ERŽEN**

Naslov: **ZASNOVA PRIPOROČILNEGA SISTEMA ZA PODPORO IZBORA ŠTUDIJSKIH PREDMETOV**

**A PROTOTYPE OF RECOMMENDATION SYSTEM FOR HIGHER EDUCATION COURSE SELECTION**

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

V diplomski nalogi preučite možnost implementacije priporočilnega sistema, ki bi študentom lahko svetoval pri izboru izbirnih predmetov. Za opis študenta naj sistem upošteva njegov predmetnik v že opravljenih letnikih. V nalogi primerjajte različne pristope k priporočilnim sistemom in poročajte o tem, kateri bi bil za dano nalogo najuspešnejši. Poročajte o eksperimentalnih rezultatih, ki jih dobite na podatkih o izboru predmetov na Visoki strokovni šoli Fakultete za računalništvo in informatiko v Ljubljani.

Mentor:

prof. dr. Blaž Zupan

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

*Besedilo je oblikovano z urejevalnikom besedil LaTeX.*

# IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani/-a Darja Eržen,

z vpisno številko 63060049,

sem avtor/-ica diplomskega dela z naslovom:

Zasnova priporočilnega sistema za podporo izbora študijskih predmetov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom
  prof. dr. Blaža Zupana
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 25.5.2012 Podpis avtorja/-ice:

# Zahvala

Najprej bi se rada zahvalila mentorju prof. dr. Blažu Zupanu za vodenje in usmerjanje pri pisanju diplomske naloge. Posebna zahvala gre tudi staršem za finančno in moralno pomoč ter fantu Roku in sestri z družino za moralno in motivacijsko podporo, tako v času študija, kot v času pisanja diplomskega dela.

# Kazalo

# Povzetek

Po vzoru priporočilnih sistemov, ki kupcem svetujejo pri nakupih izdelkov v spletnih trgovinah, smo poskušali razviti priporočilni sistem, ki študentom svetuje pri izbiri izbirnih študijskih predmetov. Zasnovali smo dve vrsti priporočilnih sistemov. Prvi sistem temelji na iskanju ključnih oznak posameznih predmetov, ki smo jih izluščili iz opisov teh študijskih predmetov. Študentu na ta način priporočamo predmete iz področij predmetov, ki jih je v preteklosti že obiskoval. Drugi sistem pa temelji na t.i. skupinskem filtriranju, kjer iz množice podatkov o preteklih izbirah predmetov iščemo vzorce izbiranja predmetov. Študentom poiščemo podobne starejše kolege in jim priporočamo predmete, ki so jih ti kolegi obiskovali v preteklih študijskih letih. Pri obeh sistemih smo uporabili različne mere za iskanje podobnosti med predmeti oziroma študenti. Na koncu smo opisana sistema tudi vrednotili in ugotovili, da je sistem, ki temelji na skupinskemu filtriranju pri svojih priporočilih uspešnejši kot sistem odkrivanja znanj iz besedilnih opisov predmetov. K temu najverjetneje botruje dejstvo, da študenti predmete izbirajo tipično raznoliko in ne vedno le v okviru enega ali dveh sorodnih področij.

## Ključne besede:

priporočilni sistem, odkrivanje znanj iz besedilnih opisov, skupinsko filtriranje, TF-IDF uteži, Jaccard-ov koeficient podobnosti, Dice-ov koeficient, kosinusna podobnost, lematizacija

# Abstract

The primary goal of this was to develop a prototype of a course recommendation system to provide decision support for the students and help them in their choice of curricula. In this, we imitated a standard on-line recommender system that serves customers by recommending them what product to buy. Two different kinds of recommendation systems were conceived. The first kind was based on mining of the textual description of the courses. Here, the students are advised to take courses whose descriptions are similar. As an alternative, we have studied recommendation systems based on collaborative filtering by proposing a selection of courses considered by students with similar course profiles. Different recommendation systems that were prototyped in our work were thoroughly evaluated on the data of course selected by students from the Faculty of Computer and Information Science. The study shows that system based on collaborative filtering gives more accurate advises than text mining system. This is likely due to the fact that students like to visit courses from various fields.

## Key words:

recommender system, text mining, collaborative filtering, TF-IDF weights, Jaccard similarity coefficient, Dice's coefficient, cosine similarity, lemmatisation

# Poglavje 1

# Uvod

V današnji informacijski, potrošniški družbi nas mediji ter ponudniki takšnih in drugačnih storitev in produktov praktično vsak dan zasipajo s svojimi ponudbami in goro informacij. V tej poplavi podatkov se le težko znajdemo in izločimo za nas relevantne informacije.

Dobro sito za ločevanje pomembnih od nepomembnih informacij je zagotovo priporočilni sistem (*angl. recommender system*), ki želi potrošnika s svojimi priporočili oz. ponudbami pridobiti na svojo stran. Tovrstne sisteme danes srečujemo predvsem na internetnih straneh, kjer spletnim brskalcem ponujajo bodisi nakup nove knjige ali ogled najnovejšega filma ali pa celo izbiro bodočega življenjskega sopotnika [3].

Koncept priporočilnih sistemov smo v tej diplomski nalogi hoteli prenesti tudi na sistem za priporočanje ali bolje rečeno "svetovanje", ki bi ugodilo predvsem študentom. Ljubljanska univerza je sedaj že dodobra zaplavala v nov, prenovljen sistem študija, ki ga narekuje bolonjska reforma. Ena glavnih značilnosti reforme je zagotovo prosto oblikovanje predmetnika glede na želje in interese posameznega študenta. Študent bo lahko svoja obzorja širil v poljubno smer, pri tem pa bo lahko iz matične fakultete pokukal tudi k sosedu.

Reforma očitno želi izboljšati študij posameznika, saj bo tako vsak študent res študiral v okviru svojih interesov in zmožnosti. Vse lepo in prav, vendar koliko napora lahko bodoči diplomant nameni, poleg trdega študija, še v izbiranje predmetov, ki so mu na voljo. Pregledovanje predmetnikov vseh 23-ih fakultet in 3-eh akademij ljubljanske univerze bi bil precej velik zalogaj, kajne?

S podobnimi problemi so se srečevali tudi na tujih univerzah. Problem so reševali s posebnimi svetovalci, največkrat profesorji, ali pa so študentje za usmeritev poprosili svoje starejše kolege. Ti pa se niso vedno izkazali kot najbolj učinkoviti, saj tudi sami niso imeli celovitega pregleda nad vsemi izbir-

nimi predmeti. Začeli so se pojavljati spletni portali, forumi in spletne učilnice, kjer so bili zbrani vsi predmeti, ki so jih fakultete tamkajšnje univerze ponujale. Študentje so na teh spletnih straneh lahko brskali med predmeti in jih tudi sami ocenjevali in komentirali[1]. Spletne strani so bile odlična podlaga za razvoj t.i. priporočilnih sistemov, ki so študentom bolj ali manj uspešno priporočali, katere študijske predmete je vredno obiskati.

Ker se tudi Univerza v Ljubljani giblje v smeri tujih univerz, bi bil podoben sistem verjetno koristen tudi za njene študente. Odločili smo se, da bomo naredili prvi korak k realizaciji takega priporočilnega sistema. Raziskali bomo, kakšni bi bili najboljši temelji za tovrsten priporočilni sistem:

1. Ali je bolje upoštevati karakteristike posameznih predmetov in jih primerjati z interesi študentov - to je t.i. priporočanje na podlagi vsebinskega izbiranja (*angl. content-based recommendation*)?

2. Ali je bolje iskati podobnosti med študenti oziroma predmeti in na podlagi le-teh priporočati ustrezno izbiro – to je t.i. skupinsko filtriranje (*angl. collaborative filtering*)?

V diplomski nalogi smo se torej lotili razvoja in vrednotenja dveh vrst priporočilnega sistema. Prvi način temelji na odkrivanju znanj iz opisov študijskih predmetov. Iskali smo ključne besede znotraj besedil, ki opisujejo predmete in na podlagi le teh iskali podobnosti med njimi. Študentu tako priporočamo izbor predmeta, ki sledi njegovim dosedanjim interesom. Drugi način pa temelji na iskanju podobnosti med študenti in podobnosti med študijskimi predmeti. Tako določenemu študentu priporočamo predmete, ki so jih obiskovali tudi starejši študentje, kateri so izbirali podobno kot opazovani študent (recimo jim podobni študentje). Oba sistema na koncu skušamo ovrednotiti na podlagi podatkov o izbiri predmetov študentov na visokošolskem študiju Fakultete za računalništvo in informatiko v študijskih letih 2010/2011 in 2011/2012.

[1]The Netflix Effect: When Software Suggests Students' Courses.
Dostopno na: `http://chronicle.com/article/article-content/127059/`

## 1.1 Motivacija za razvoj priporočilnega sistema za priporočanje pri izbiri izbirnih predmetov

Z novo bolonjsko reformo se je študij na naših univerzah kar precej spremenil. Pozitivna stran reforme je zagotovo širok izbor izbirnih predmetov, ki so na voljo študentom. Izbor predmetov pa se z uvajanjem novih študijskih programov še dodatno širi in bodoči diplomanti se včasih težko odločijo, katere predmete bi bilo najbolje obiskovati. S še večjimi težavami se srečujejo novi študentje, bodisi tuji študentje na izmenjavi ali pa študentje prvih letnikov, ki še ne poznajo veliko starejših kolegov in profesorjev na fakulteti. Da bi bile težave pred in tudi med študijem študentom čim bolj prihranjene, so se na spletu začeli pojavljati razni portali za ocenjevanje predmetov in profesorjev, na spletnih forumih so začeli ljudje spraševati in komentirati razne smeri študija ipd. Tudi fakultete same so po večini poskrbele za čim bolj udoben študij s svojimi spletnimi učilnicami.

Kot vsaka konkurenčna spletna trgovina, ki želi na svojo stran privabiti čim več novih in ohraniti čim več starih kupcev, mora tudi fakulteta (oz. celotna univerza) poskrbeti, da zanimanje zanjo ne bo začelo upadati. Tako kot gigant na področju spletne prodaje Amazon.com je svojo prodajo pospešil predvsem s svojimi priporočili spletnim kupcem, bi lahko tudi v visokošolskem sistemu na podoben način novim in starim študentom priporočali študijske predmete in dodatne dejavnosti, ki se izvajajo na posameznih fakultetah.

Na univerzah po svetu se dobro zavedajo, da se tudi v šolstvu pojavlja konkurenca, ki zadnja leta z izmenjavami študentov krepko presega tudi državne meje. Nekatere tuje univerze so tako nadgradile svoje portale in spletne učilnice s planerji urnikov in ponekod pa tudi s svetovalnimi sistemi, ki pomagajo uporabnikom pri izbiri študijskih predmetov.

Na univerzi v Pittsburgu (School of Information Sciences, University of Pittsburgh) so tako razvili poseben sistem imenovan *Course Agent*, ki je namenjen planiranju predmetnika posameznega študenta. Njegova priporočila in svetovanje sta predvsem usmerjena v študentove cilje za njegovo bodočo kariero. Priporočila sistema temeljijo na eksplicitnih informacijah, ki mu jih podaja študent tekom svojega študija [1].

Tudi na Norveškem so se lotili podobnega projekta. Na fakulteti Norwegian University of Science and Technology so zasnovali posebno spletno stran imenovano *Classmate*. S pomočjo strani naj bi zbirali podatke o študentih, njihove cilje, interese in seveda izbire študijskih predmetov. Razvili so hibridni

sistem, ki temelji tako na skupinskem filtriranju podatkov (o izbiri predmetov), kot tudi na priporočanju na podlagi vsebinskega izbiranja. Med dejavniki, ki vplivajo na izbiro študijske smeri, so upoštevali tako klasične podatke o predmetih (letnik, semester, način dela in ocenjevanja, predpogoji,...), kot tudi osebne podatke o študentih (želje za kariero, interesi, pretekla znanja,...). Na spletni strani Classmate bodo študentje urejali svoje urnike in tudi ocenjevali in komentirali delo profesorjev in izbranih predmetov [2].

Hitremu razvoju in novostim v gospodarstvu očitno sledijo tudi izobraževalne ustanove po svetu. Tudi naša univerza bi lahko po vzoru norveške ali ameriške univerze začela z večjim povezovanjem svojih fakultet in akademij in tako študentom omogočila čim bolj poučen in raznolik študij, bodisi na eni ali pa na več njenih ustanovah.

# Poglavje 2

# Metode in pristopi

## 2.1 Priprava podatkov

### 2.1.1 Kratka predstavitev predmetnika visokošolskega študija na FRI

Novi program visokošolskega študija na Fakulteti za računalništvo in informatiko je bil po bolonjski reformi vzpostavljen pred tremi leti. Študij traja 3 leta oziroma 6 semestrov. Prvo leto oziroma prva dva semestra študenti obiskujejo 10 obveznih predmetov, ki študentom zagotovijo ustrezna osnovna matematična, teoretična in strokovna znanja. V naslednjih treh semestrih pa študent sam izbira med t.i. izbirnimi predmeti, ki njegovo znanje usmerijo na različna področja. Zadnji semester v 3. letniku je rezerviran za obvezno prakso v podjetju in izdelavo diplomske naloge[1].

Kot sem že omenili, si v 2. in 3. letniku študent predmetnik sestavi sam s poljubno izbiro predmetov. Pri tem pa mora upoštevati določene omejitve:

1. pri izbiri predmeta mora študent upoštevati predpogoje (ti so na sliki 2.1 označeni s puščicami oziroma z zvezdico) in

2. študent mora imeti ob zaključku študija opravljena vsaj dva izmed "ciljnih" predmetov (na sliki 2.1 so pobarvani rdeče), ki predstavljata različni področji računalništva.

Študenta naj bi izbran predmetnik pripeljal do globljega znanja vsaj dveh izmed različnih računalniških področij: spletne tehnologije, programska oprema,

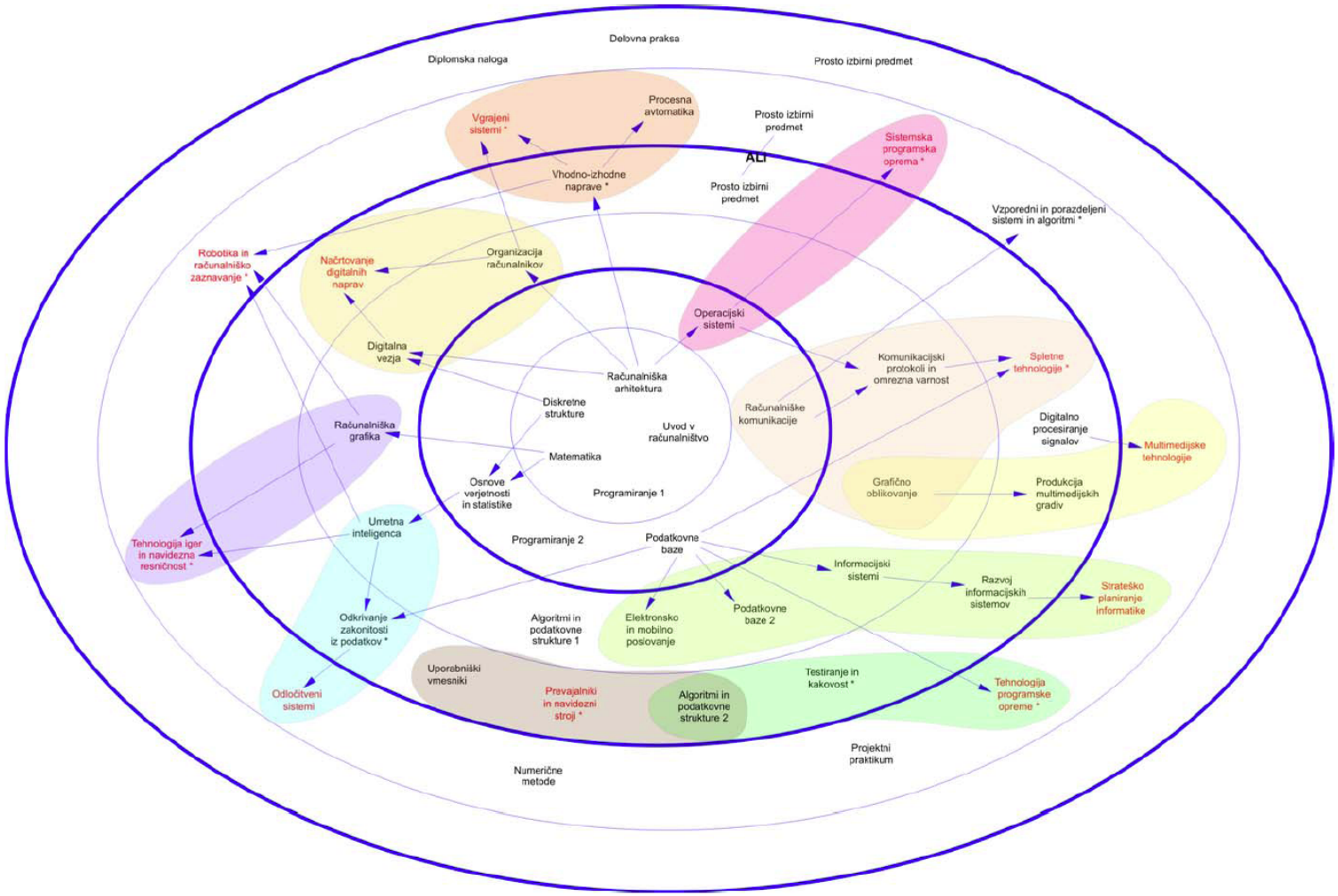
[1]Predstavitveni zbornik FRI-VSŠ. Dostopno na: `http://www.fri.uni-lj.si/file/113229/predstavitveni-zbornik-fri-vss.pdf`

informatika, tehnologija znanja, sistemska administracija, inteligentni sistemi, multimedija, procesna informatika, računalniško inženirstvo, robotika, procesna avtomatika[2]...

Študijski program je sestavljen iz skupno 43-ih predmetov. V prvem letniku študent opravi 10 obveznih predmetov. V drugem letniku izbira med 21-imi predmeti (10 v zimskem in 11 predmetov v poletnem semestru), 5 jih izbere v zimskem in 5 v poletnem semestru. V tretjem letniku študent opravlja en obvezni predmet, ostale pa izbira iz nabora 11-ih predmetov 3. letnika in nabora neizbranih predmetov iz zimskega semestra 2. letnika. Tekom študija mora študent izbrati še 2 prosta izbirna predmeta, in sicer v 4. ali 5. semestru oziroma v 5. ali 6. semestru. Prosti izbirni predmeti so lahko tudi predmeti, ki se izvajajo izven fakultete. Zadnji semester študenti zaključijo z 9-tedensko prakso in izdelavo diplomske naloge.

Študenti imajo torej pri oblikovanju svojega predmetnika dokaj proste roke. Upoštevati morajo le določene omejitve in v vsakem semestru izbrati predmete v obsegu 30 kreditnih točk.

[2]Visokošolski strokovni študij Računalništvo in informatika - Opis programa. Dostopno na: `http://www.fri.uni-lj.si/si/visokosolski\_strokovni/opis/`




Slika 2.1: Sistem povezav med študijskimi predmeti: predpogoji predmetom so ponazorjeni s puščicami, predmeti označeni z zvezdico imajo za predpogoj še predmet APS1. Rdeče obarvani predmeti so ciljni predmeti. Določena računalniška področja, ki jih tvorijo različne skupine predmetov, so obarvana z različnimi barvami.

### 2.1.2 Priprava in strukturiranje podatkov

Pred vsako implementacijo priporočilnega sistema je potrebno pripraviti ustrezne podatke. Priprava podatkov obsega zajem pravih podatkov, bodisi iz neke vnaprej pripravljene podatkovne baze ali pa podatke iščemo avtomatsko po besedilu ipd. Zajemu podatkov ponavadi sledi krajša analiza in urejanje le-teh in nato še ustrezno strukturiranje podatkov v primerno obliko.

Za našo pripravo priporočilnega sistema bomo potrebovali splošne podatke o predmetih. Ti zajemajo naslednje atribute:

1. identifikacijska oznaka predmeta,
2. naziv predmeta,
3. letnik, v katerem se predmet izvaja (2. ali 3.),
4. semester, v katerem se predmet izvaja (zimski ali poletni),
5. predpogoji za opravljanje predmeta (identifikacijske oznake predmetov, ki jih mora študent predhodno oz. vzporedno opravljati),
6. podatek o tem, ali je predmet ciljni.

Podatke o predmetih smo pripravili sami, in sicer na podlagi informacij, ki smo jih pridobili na spletni strani Fakultete za računalništvo in informatiko `http://www.fri.uni-lj.si/`. Podatke smo shranili v tekstovno datoteko, in sicer tako, da so vrednosti atributov med seboj ločene s tabulatorji.

Poleg podatkov o predmetih smo iz študijskega informacijskega sistema fakultete (e-Študent) pridobili še podatke o študentih in njihovih izbranih izbirnih predmetih v študijskih letih 2010/2011 in 2011/2012. Podatki so zajemali identifikacijske številke predmetov skupaj z nazivom predmetov, vpisne številke študentov, ki so bile predhodno ustrezno spremenjene (zamenjane z naključnimi števili, da študenti ostanejo anonimni), študijsko leto, letnik študenta, smer študija in kraj izvajanja študija (Ljubljana ali Sežana). Te podatke smo po krajši analizi ustrezno uredili in odstranili vse nepravilnosti.

Priprava podatkov, ki smo jih pridobili iz sistema e-Študent, je vzela kar nekaj časa. Podatke smo najprej prebrali iz klasične tekstovne datoteke v podatkovno bazo MySQL za lažjo obdelavo. Med podatki je bilo kar nekaj nejasnosti (npr. študent je izbral predmet iz 3. letnika v 2. letniku ali pa je študent izbral predmet za katerega ni imel ustreznih pogojev itn.). Pri urejanju podatkov smo si pomagali s SQL poizvedbami in PHP programskim jezikom. Za konec smo vse podatke shranili v klasično tekstovno datoteko, tako da so

bile posamezne vrednosti atributov ločene s tabulatorjem. V datoteko smo shranili le podatke, ki so za nas pomembni:

1. identifikacijska številka študenta
2. letnik, v katerem je študent predmet izbral
3. identifikacijska oznaka izbranega predmeta

Za potrebe odkrivanja znanj iz besedilnih opisov predmetov so potrebni še opisi posameznih izbirnih predmetov. Te opise smo pridobili na spletni strani fakultete[3]. Opise in ostale podatke o predmetih smo shranili v tekstovne datoteke (vsak predmet v svojo datoteko). Te tekstovne datoteke smo nato procesirali s pomočjo sistema za lematizacijo, kot je opisano v poglavju 2.2.1 in tako pridobili ustrezne sezname besed, ki smo jih v nadaljevanju potrebovali pri pripravi korpusa uteženih ključnih besed posameznih predmetov.

## 2.2 Odkrivanje znanj iz besedilnih opisov

Odkrivanje znanj iz besedil (*angl. text mining*) je na področju umetne inteligence ena od vej odkrivanja znanj iz podatkovnih baz - podatkovnega rudarjenja (*angl. data mining*). Gre za analiziranje besedil, ki se pogosto uporablja pri priporočilnih sistemih – natančneje pri *priporočanju na podlagi vsebinskega izbiranja* (*angl. content-based recommendation*). Najbolj tipičen predstavnik tovrstnega priporočilnega sistema je zagotovo priporočanje spletnih člankov, pri katerem sistem bralcu ponudi članek oziroma besedilo, ki bralca utegne zanimati. Sistem primerja ključne besede člankov, ki jih je bralec v preteklosti že prebral, s ključnimi besedami nekih novih, domnevno neprebranih člankov [3].

Tehnike odkrivanja znanj iz besedil bomo uporabili pri iskanju ključnih besed za posamezen študijski predmet. Ključne besede in besedne zveze bomo skušali izluščiti iz besedil oziroma bolje rečeno opisov predmetov, ki smo jih predhodno pripravili.

S pomočjo ključnih besed bomo lahko študijske predmete med seboj primerjali (iskali podobnosti med njimi) in jih priporočali študentom v skladu z njihovimi interesi.

[3]Vsebine predmetov VŠ. Dostopno na:
`www.fri.uni-lj.si/file/113863/vsebine-predmetov-vs.pdf`

### 2.2.1 Lematizacija

Lematizacija je postopek določanja t.i. leme oziroma osnovne (slovnične) oblike posameznim besedam v besedilu, npr. besede "pišeš", "pišemo", "pisali" imajo skupno lemo "pisati" [4].

S pomočjo lematizacije bomo vse besede iz opisov "poenostavili" in tako omogočili bolj natančno primerjavo. Poglejmo si primer dveh povedi iz različnih opisov študijskih predmetov:

A) Produkcija multimedijskih gradiv:

> *"Cilj predmeta je študente spoznati z različnimi vidiki produkcije multimedijskih vsebin."*

B) Multimedijske tehnologije:

> *"Vaje v primerno opremljenih študentskih laboratorijih, kjer so na voljo naprave za delo z multimedijskimi vsebinami."*

Ključna beseda oziroma besedna zveza teh dveh povedih je zagotovo (poleg drugih) "multimedijska vsebina". Vendar se ta beseda v obeh povedih pojavi v različnih slovničnih oblikah in bi ju računalnik seveda dojemal kot dve popolnoma različni besedi: "multimedijskih vsebin" $\neq$ "multimedijskimi vsebinami".

Primerjava izvirnih besed iz opisov se torej ne bi obnesla. Zato uporabimo t.i. postopek lematizacije, ki vsako od besed spremeni v njeno osnovno obliko in jo tako pripravi na učinkovitejšo računalniško primerjavo.

Uporabili bomo že izdelan sistem za lematizacijo (lematizator), ki je na voljo na spletni strani Inštituta Jožef Stefan (`http://nl2.ijs.si/analyze/`). Kot je razloženo v navodilih na spletni strani, je vhod lematizatorja tekstovna datoteka (kodirana po UTF-8). Lematizator nam po procesiranju vhodne tekstovne datoteke vrne izhodno tekstovno datoteko (s končnico .ske), v kateri so zapisane ustrezne osnovne oblike oziroma leme za vsako besedo iz podanega besedila.

V tabeli 2.1 je prikazan primer lematizacije ene od povedi iz opisa študijskega predmeta Umetna inteligenca:

> *"Uporaba predstavljenih metod na konkretnih problemih iz znanstvenega in poslovnega okolja."*

Tabela 2.1: Prikaz izhoda iz sistema za lematizacijo besed.

| | | |
|---|---|---|
| Uporaba | Sozei | uporaba |
| predstavljenih | Pdnzmr | predstavljen |
| metod | Sozmr | metoda |
| na | Dm | na |
| konkretnih | Ppnmmm | konkreten |
| problemih | Sommm | problem |
| iz | Dr | iz |
| znanstvenega | Ppnser | znanstven |
| in | Vp | in |
| poslovnega | Ppnser | posloven |
| okolja | Soser | okolje |

Prvi stolpec pri izhodni datoteki (glej tabelo 2.1) predstavlja originalno besedo iz opisa. V drugem stolpcu je oznaka besedne vrste oziroma morfološki opis za posamezno besedo, npr. oznaka *Dm* pomeni "predlog, sklon=mestnik" (pomen vsake od oznak je razložen na spletni strani inštituta `http://nl.ijs.si/jos/msd/html-en/msd.index.msds.html`). V tretjem stolpcu pa najdemo lemo besede iz prvega stolpca. Lematizator upošteva prav vse znake v besedilu, tudi npr. ločila, nove vrstice ipd. Take nepomembne podatke pri obravnavi lematiziranih besedil seveda izpustimo.

### 2.2.2 Iskanje ključnih besed in besednih zvez

Pri iskanju ključnih besed oziroma oznak predmetov se bomo omejili le na samostalniške in pridevniške besede. Ostale besedne vrste, kot so veznik, členek, predlog in druge predstavljajo namreč tako imenovane manj pomembne besede, ki ne nosijo nekega relevantnega pomena (*angl. stop-words*). Pri obravnavi besed pa bomo izpustili tudi glagole, ki po navadi niso uporabljeni pri označevanju besedil (npr. spletnih člankov) oziroma v našem primeru študijskih predmetov.

Pri opisih predmetov se pojavi veliko (pomembnih) besednih zvez (dve ali več pomensko in oblikovno povezanih besed), kot so npr. *informacijski sistem*, *podatkovna baza*, *umetna inteligenca*, *strojno učenje*, *relacijska baza*, ipd. Ker bi besedne zveze najverjetneje prispevale k boljši primerjavi predmetov, bomo tudi te skušali vključiti pri primerjavi predmetov.

Algoritem za iskanje ključnih besed in besednih zvez najprej prebere izhodno datoteko, ki jo je sestavil lematizator. V podatkovno strukturo slovar shranimo vse besede in njihove oznake besednih vrst. V naslednjem koraku algoritem iz slovarja bere besedo za besedo. Upošteva le pridevniške in samostalniške besede in ob enem tudi sestavlja besedne zveze (pridevniška beseda skupaj s samostalniško besedo tvori besedno zvezo, primer: *informacijski + sistem*, *strojno + učenje*). Opisan postopek izločevanja oznak se ponovi za vsak opis posameznega študijskega predmeta. Na koncu algoritem vrne korpus besed oziroma gnezdeni slovar, kjer imamo zbrane ključne besede in besedne zveze za vsak predmet posebej skupaj s številom ponovitev te besede pri posameznem predmetu. V slovarju pa je shranjen tudi seznam predmetov, kjer se posamezna beseda pojavi. Da pa korpus ne bi bil preobsežen, algoritem pred koncem izloči vse besede, ki so se pojavile le pri enem od opisov predmeta. Opisani algoritem lahko vidimo na sliki 2.2.

Poglejmo si primer iskanja ključnih besed in besednih zvez v eni od povedi:

> *"Pri predmetu Umetna inteligenca boste spoznali, zakaj obstaja tako velik razkorak med pričakovanji in realnostjo, kaj sploh je inteligenca, in kako jo simuliramo z računalnikom."*

Izločene besede in besedne zveze v zgornji povedi so: *predmet, umeten inteligenca, velik razkorak, pričakovanje, realnost, inteligenca, računalnik.*

Kot lahko opazimo besedne zveze niso napisane v pravilni, slovnični obliki (*"umeten inteligenca"* namesto *"umetna inteligenca"*). To je posledica lematizacije, ki za vsako besedo določi lemo in tako vse pridevnike spremeni v moško obliko ednine (sklon v imenovalniku). Pri primerjavi oblika besede ni pomembna. V kolikor pa želimo oznake prikazovati končnemu uporabniku, pa bi bilo potrebo narediti še poseben slovar, kjer bi bile besedne zveze zapisane v pravi obliki (pravi sklon, spol in število). Samostojne besede pri tem niso problematične.

Na tem mestu lahko omenimo še, da je možno iskati tudi besedne zveze iz treh besed (npr. 2 pridevniški skupaj s samostalniško besedo), kot je npr. *sistemska programska oprema* ali pa *sodoben informacijski sistem*. Vendar smo pri testiranju ugotovili, da se tovrstne oznake ne izkažejo za najboljše, ker je ključna beseda oziroma njeno bistvo, preveč "skrita"; npr. besedna zveza *informacijski sistem* se pojavi kot *sodoben informacijski sistem*, *podprt informacijski sistem*, ... Konec koncev pa je skupna točka tema dvema besednima zvezama kar sama oznaka *informacijski sistem*. Zato smo se odločili, da bomo uporabili le samostojne ključne besede in ključne besedne zveze iz dveh besed.

**Algorithm 2.2.1:** CONSTRUCTCORPUS(*lematizedFileList*)

```
subjectsList ← GETSUBJECTSLIST()
corpus ← {}
lists ← {}

for subjects ∈ subjectsList :
keywordsList ← []
while read(lematizedFileList[subject])! = −1 :
    do  if word[i] = noun and word[i + 1] = adjective :
            then { keywordsList[] ← word[i] + word[i + 1]
                 { lists[word[i] + word[i + 1]] ← subject

        else if word[i] = noun or word[i] = adjective :
            then { keywordsList[] ← word[i]
                 { lists[word[i]] ← subject

        else continue;

corpus[subject] ← keywordsList
corpus[lists] ← lists

corpus ← DELETEONETIMEWORDS(corpus)
return (corpus)
```

Slika 2.2: Priprava korpusa ključnih besed za izračun uteži teh besed in kasnejšo uporabo pri odkrivanju znanj iz besedilnih opisov.

### 2.2.3 Določanje uteži besedam in besednim zvezam

Vsaki od ključnih besed (oznak) bomo določili utež, s katero izrazimo njeno pomembnost. Oznake, kot so npr. *študent*, *predmet*, *predavanje*, *izpit* se pojavijo praktično pri vseh predmetih in nam pri primerjavi predmetov ne koristijo kaj dosti. Na drugi strani pa so oznake kot npr. odločitven sistem, podatkovna struktura, mikroprocesorski sistem, ki se pojavijo v le nekaj opisih predmetov (in pri teh opisih večkrat) in so ključne oznake tega študijskega predmeta. Take oznake morajo imeti seveda večjo težo.

Pomembnost posameznih ključnih besed bomo določali s t.i. TF-IDF (*angl. Term frequency – inverse document frequency*) utežmi. Te uteži so pogosto uporabljene pri informacijskih poizvedbah (*angl. information retrieval*) in odkrivanju znanj iz besedil (*angl. text mining*). Pomembnost oznake se povečuje s številom ponovitev besed v določenem besedilu in se ob enem izravna s številom ponovitev besed v vseh besedilih. Torej beseda, ki se velikokrat ponovi v nekem opisu in se ne pojavi pogosto v drugih opisih, je za opazovano besedilo lahko zelo pomembna in je njena teža seveda večja[4] [6].

Izračun TF-IDF uteži je preprost. Prvi del - TF (*angl. term frequency*), prikazan v enačbi (2.1), predstavlja frekvenco oznake i v besedilu j. Izračunamo jo kot razmerje med številom ponovitev oznake i v besedilu j in številom vseh oznak M v besedilu j [3].

$$TF_{ij} = \frac{freq_{ij}}{M_j} \tag{2.1}$$

Drugi del izračuna uteži je IDF (*angl. inverse document frequency*) - enačba (2.2). To inverzno funkcijo izračunamo kot logaritem (npr. naravni logaritem z osnovo e) razmerja med številom vseh dokumentov N in številom dokumentov v katerih se pojavi opazovana oznaka i [3].

$$IDF_i = ln\frac{N}{n_i} \tag{2.2}$$

Utež pa izračunamo s preprostim produktom prvega in drugega dela, kot je prikazano v enačbi (2.3).

$$TF - IDF_{i,j} = TF_{ij} \times IDF_i \tag{2.3}$$

V naslednjih primerih si lahko pogledamo primer dveh ključnih besed - pomembno oznako *"strojen učenje"* in manj pomembno oznako *"seminar"*.

[4] tf*idf. Dostopno na: `http://en.wikipedia.org/wiki/TF_IDF`

Oznaka *"strojen učenje"* se pri opisu predmeta Umetna inteligenca pojavi 3-krat. Opis je sestavljen iz 174 oznak. Vseh dokumentov je 32. Oznaka *"strojen učenje"* se pojavi v skupno dveh dokumentih.

$$TF - IDF_{strojenucenje,UI.txt} = \frac{3}{174} \times ln(\frac{32}{2}) = 0.0478$$

Oznaka *"seminar"* se pri istem opisu pojavi le 1-krat, vendar se pojavi še v šestih (skupno sedmih) drugih opisih.

$$TF - IDF_{seminar,UI.txt} = \frac{1}{174} \times ln(\frac{32}{7}) = 0.00873$$

Čeprav se nobena od oznak ne pojavi prav velikokrat pri opisu predmeta Umetna inteligenca, je razlika med njunima utežema kar precejšna (petkratna). Razlika med oznakama je, da se oznaka *"strojen učenje"* pojavi le v opazovanem in še enem opisu, medtem ko se oznaka *"seminar"* pojavi v kar sedmih različnih opisih in je tako bolj vsakdanja in posledično manj pomembna. Besede, kot so "predavanja", "predmet", "študent" se pojavijo v vseh opisih in je njihova utež zato enaka 0 ($IDF = ln(1) = 0$).

Na koncu računanja uteži še vse TF-IDF uteži dodatno normaliziramo, kot je prikazano v enačbi (2.4). Z normalizacijo tako izničimo vpliv različnih dolžin opisov predmetov. Normaliziramo tako, da vsako utež i delimo s korenom seštevka kvadratov vseh TF-IDF uteži, ki jih ima M ključnih oznak v opazovanem besedilu j [6].

$$w_{ij} = \frac{TF - IDF_{i,j}}{\sqrt{\sum_{s=1}^{M}(TF - IDF_{s,j})^2}} \tag{2.4}$$

### 2.2.4 Priprava in uporaba korpusa ključnih besed

Kot smo že omenili v poglavju 2.2.2, ključne besede in besedne zveze preberemo iz datotek, ki nam jih je vrnil lematizator in jih shranimo v gnezdeni slovar skupaj z njihovimi frekvencami. Po izračunu TF-IDF uteži za vsako od teh oznak pri vseh opisih predmetov konstruiramo korpus - gnezdeni slovar, ki za vsak predmet hrani urejen seznam oznak (ključnih besed in besednih zvez) skupaj z njihovimi utežmi. Seznam je urejen od oznake z največjo utežjo (najpomembnejšo) do oznake z najmanjšo težo (najmanj pomembno). Urejenost oznak je pomembna, ker ob računanju podobnosti med predmeti po navadi zajamemo le neki delež oznak, ki imajo največje uteži - torej preprosto zajamemo le "vrh" seznama oznak.

Kot primer si poglejmo tri predmete iz različnih področij računalništva in njihovih petih najpomembnejših oznak glede TF-IDF uteži:

A) Odkrivanje zakonitosti iz podatkov: *odkrivanje, zakonitost, podatek, vzorec, strojen učenje.*

B) Vhodno-izhodne naprave: *naprava, zgradba, digitalen vezje, računalniški arhitektura, lastnost.*

C) Razvoj informacijskih sistemov: *informacijski sistem, metodologija, zahteva, razvoj, sistematičen pristop.*

Korpus ključnih besed smo v našem primeru shranili kot gnezdeni slovar, saj naš korpus zajema le 32 različnih študijskih predmetov. Pri obsežnejših zbirkah izbirnih predmetov pa bi bilo bolj smiselno korpus oznak z utežmi hraniti kar v primerni podatkovni bazi.

### 2.2.5 Računanje podobnosti med opisi študijskih predmetov

Tako kot spletnemu brskalcu priporočilni sistem priporoča nove neprebrane članke, ki so sorodni (podobni) že prebranim besedilom, bomo tudi mi skušali študentu predlagati nove predmete, ki so sorodni predmetom, ki jih je študent v preteklosti že obiskoval. Računali bomo torej podobnost med predmeti glede na njihove ključne oznake, ki smo jih pridobili s prej opisanimi metodami in algoritmi.

Ena od metod za računanje podobnosti med besedili je Jaccard-ov koeficient podobnosti (*angl. Jaccard similarity coefficient*), ki računa podobnost med množicama različnih vrednosti [5] (v našem primeru so množice vrednosti množice oznak predmetov shranjene v korpusu, ki smo ga konstruirali). Koeficient je definiran kot razmerje med velikostjo preseka množic A in B in velikostjo unije teh dveh množic, kot je prikazano v enačbi (2.5).

$$J(A, B) = \frac{\mid A \cap B \mid}{\mid A \cup B \mid} \tag{2.5}$$

Druga metoda za računanje podobnosti je Dice-ov koeficient (*angl. Dice's coefficient*), ki je pravzaprav precej podoben Jaccard-ovemu. Dice-ov koeficient izračunamo kot razmerje med dvakratno velikostjo preseka množic oznak A in B in med seštevkom moči teh dveh množic [5], kot je prikazano v enačbi (2.6).

$$D(A, B) = \frac{2\times \mid A \cap B \mid}{\mid A \mid + \mid B \mid} \tag{2.6}$$

Pri obeh zgoraj opisanih metod uporabimo zgolj ključne oznake iz korpusa, ne pa njihovih TF-IDF uteži. Zato bomo za računanje podobnosti uporabili še tretjo metodo, ki je pravzaprav najbolj pogosto uporabljena metoda pri računanju podobnosti dveh množic oziroma vektorjev uteži. To je kosinusna razdalja oziroma podobnost (*angl. cosine similarity*)[5]. Pri množenju in seštevanju uteži v "kosinusni enačbi" (2.7) upoštevamo le skupne oznake dveh primerjanih predmetov z oznakami v množicah A in B (recimo, da so te skupne oznake v množici $C = A \cap B$):

$$sim(A, B) = \frac{\sum_c w_{A_c} \times \ w_{B_c}}{\sqrt{\sum_c w_{A_c}^2} \times \sqrt{\sum_c w_{B_c}^2}} \tag{2.7}$$

Kot zadnjo metodo bomo preizkusili še poenostavljeno različico računanja kosinusne podobnosti. Podobnost se izračuna s preprostim množenjem normaliziranih uteži skupnih oznak dveh predmetov [5], kot je prikazano v enačbi (2.8).

$$sim(A, B) = \sum_c w_{A_c} \times \ w_{B_c} \tag{2.8}$$

Izvorna koda (spisana v programskem jeziku Python) vseh štirih predstavljenih metod za računanje podobnosti je prikazana v dodatku D na koncu diplomske naloge (glej izvorno kodo calculateSimilarity_TM.py).

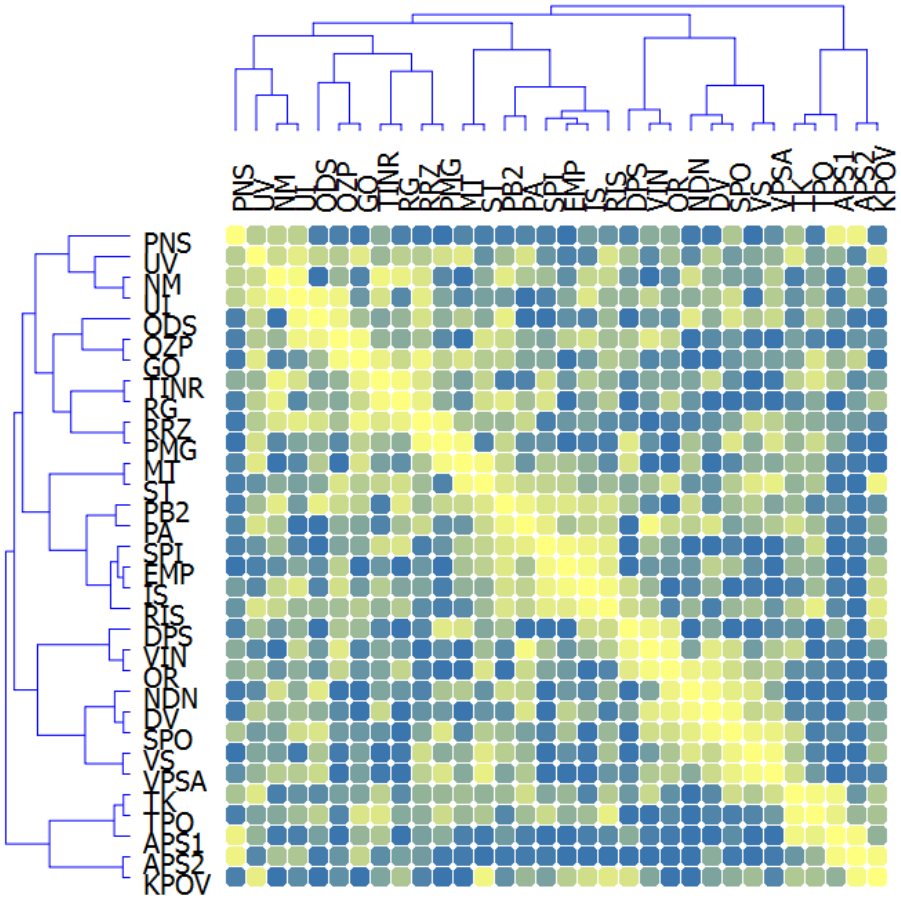
V tabeli 2.2 smo navedli primer petih najbolj podobnih študijskih predmetov predmetu *Elektronsko in mobilno poslovanje*, izračunanih po vseh štirih opisanih metodah.

Tabela 2.2: Prikaz petih najbolj podobnih predmetov predmetu *EMP* pri odkrivanju znanj iz opisov (izračunano po štirih metodah).

| **Jaccard-ov koeficient** | **Dice-ov koeficient** |
|---|---|
| Strateško planiranje informatike | Strateško planiranje informatike |
| Informacijski sistemi | Informacijski sistemi |
| Razvoj informacijskih sistemov | Razvoj informacijskih sistemov |
| Podatkovne baze 2 | Podatkovne baze 2 |
| Spletne tehnologije | Spletne tehnologije |
| **Klasična kosinusna podobnost** | **Poenostavljena kosinusna podobnost** |
| Testiranje in kakovost | Informacijski sistemi |
| Procesna avtomatika | Strateško planiranje informatike |
| Kom. protokoli in omrežna varnost | Razvoj informacijskih sistemov |
| Strateško planiranje informatike | Kom. protokoli in omrežna varnost |
| Razvoj informacijskih sistemov | Spletne tehnologije |

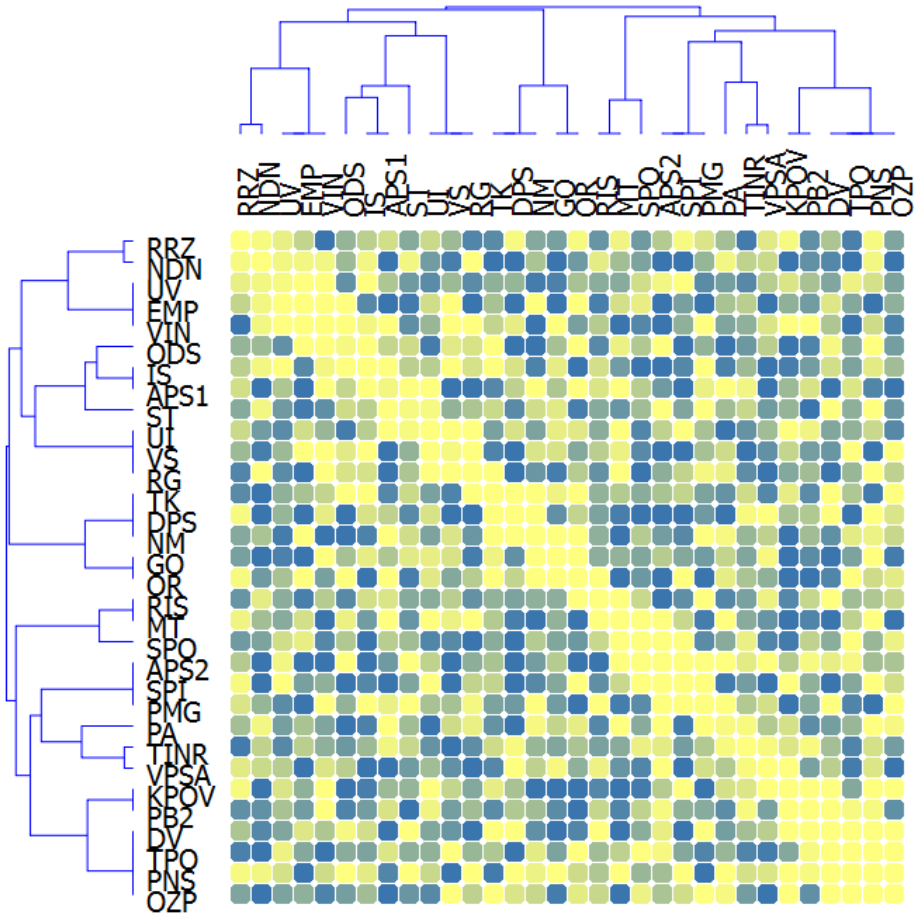
Da pa bi si lažje predstavljali v kolikšni meri so si študijski predmeti med seboj dejansko podobni, smo izračunane podatke o podobnostih med posameznimi predmeti tudi vizualizirali. Podobnosti so predstavljene s t.i. toplotno karto (*angl. heatmap*). Rumena barva na karti predstavlja veliko podobnost, modra barva pa manjšo podobnost med dvema predmetoma. Toplotne karte so izdelane s pomočjo odprtokodnega programa Orange. Za lažje pregledovanje toplotnih kart je na koncu diplomske naloge v dodatku A priložena preglednica A.1 polnih nazivov študijskih predmetov skupaj z oznakami predmetov.

Na sliki 2.3 vidimo vizualizacijo podobnosti med posameznimi predmeti, ki je bila izračunana pri odkrivanju znanj iz besedilnih opisov, in sicer s Jaccardovim koeficientom (zajetih 50% vseh ključnih oznak predmetov). Oznake predmetov v vrstici oziroma stolpcu toplotne karte so razporejeni s t.i. hierarhičnim razvrščanjem v skupine (*angl. hierarchical clustering*) glede na podobnost med predmeti. Hitro lahko opazimo, da avtomatsko razvrščanje v skupine skupaj postavi predmete iz istih področij (podobno kot so področja z različnimi barvami predstavljena na sliki 2.1 v poglavju 2.1). Skupaj tako recimo stojijo predmeti iz področja informatike (IS, RIS, SPI, EMP), področja strojne opreme (DPS, VIN, OR, NDN, DV) ipd. Zaradi te hierarhije je na diagonali (od zgornjega levega kota karte proti spodnjemu desnemu kotu karte) moč zaznati nekoliko večja območja rumenih odtenkov (rumena kaže na večjo podobnost).




Slika 2.3: Vizualizacija podobnosti med predmeti izračunana z odkrivanjem znanj iz besedilnih opisov predmetov (Jaccard-ov koeficient, zajetih 50% ključnih oznak predmetov).

Po krajšem testiranju ostalih treh metod računanja podobnosti med predmeti ugotovimo, da Dice-ov koeficient in poenostavljena kosinusna podobnost dajeta podobne rezultate kot Jaccard-ov koeficient. Avtomatična hierarhična razporeditev, ki jo določi program Orange, je skoraj identična. Diagonala je prav tako v rumenih odtenkih. Nekoliko drugačno razporeditev predmetov pa opazimo pri klasični kosinusni podobnosti (glej enačbo (2.7)), katere toplotni graf je na sliki 2.4. Tukaj predmeti niso razporejeni po področjih, kot bi pričakovali.




Slika 2.4: Vizualizacija podobnosti med predmeti izračunana z odkrivanjem znanj iz besedilnih opisov predmetov (klasična kosinusna podobnost, zajetih 50% ključnih oznak predmetov).

za zdaj lahko rečemo, da bodo tri metode (Jaccard-ov koeficient, Dice-ov koeficient, poenostavljena kosinusna podobnost) študentom vsaj po večini priporočale predmete iz sorodnih področij. In če predpostavimo, da študentje dejansko izbirajo predmete po takem ključu, se bodo te tri metode pri testiranju odrezale bolje kot klasična kosinusna podobnost. Več o uspešnosti posameznih metod pa bomo lahko povedali šele po vrednotenju posameznih sistemov v poglavju 3.

## 2.3 Skupinsko filtriranje

Skupinsko filtriranje (*angl. collaborative filtering*) je metoda, s katero lahko napovemo (izračunamo), kako bo nek kupec ocenil opazovani izdelek (storitev) [3]. Napoved temelji na preteklem obnašanju drugih kupcev, ki so podobni opazovanemu. Filtriranje se najpogosteje uporablja pri spletnih trgovinah (ena največjih spletnih trgovin, ki uporablja ta princip priporočilnih sistemov, je Amazon.com), kjer spletni ponudnik zbira podatke o svojih kupcih in njihovih navadah. Na podlagi teh podatkov potem lahko registriranemu kupcu ponuja različne izdelke, ki naj bi mu bili všeč [7]. Ponujeni izdelki so bili dobro ocenjeni s strani registriranih kupcev, ki so v preteklosti podobno ocenjevali/izbirali izdelke kot opazovan kupec.

Če želimo spremljati navade in okus naših kupcev, moramo zbirati njihove ocene kupljenih izdelkov ali pa vsaj beležiti, katere izdelke je določen kupec kupil. Tu se pojavi pomanjkljivost skupinskega filtriranja. Glavni problem pri postavitvi priporočilnega sistema, ki bi temeljil na tovrstnem filtriranju, je zagotovo problem hladnega zagona (*angl. cold start*), ki se pojavi ob začetnem zagonu sistema in ob vstopu nove stranke v sistem [3]. Ob postavitvi sistema nam pogostokrat primanjkuje podatkov. Ko novo stranko zabeležimo v sistem, ponavadi o njej oz. njenih navadah ne vemo ničesar. Zato se skupinsko filtriranje kombinira s priporočanjem na podlagi vsebinskega izbiranja (del tovrstnega priporočanja je odkrivanje znanj iz besedilnih opisov, opisano v poglavju 2.2).

Ker so se tehnike skupinskega filtriranja pri priporočanju v preteklosti izkazale kot zelo uspešne, bomo tudi mi poskusili razviti sistem, ki bo študentu priporočal predmete, ki so jih poslušali tudi njemu podobni starejši kolegi. V nasprotju z vsebinskim izbiranjem tokrat ne bomo upoštevali študentovih interesov, ampak bomo v množici študentov iskali vzorce izbiranja predmetov.

Tipično se pri skupinskem filtriranju preračunava ocene, ki so jih kupci dodelili določenim izdelkom (recimo 1 za slabo mnenje in 10 za odlično mnenje o izdelku). Ker v našem primeru na žalost nimamo podatkov o tem, kako bi (so) študentje ocenili predmete, ki so jih obiskovali, se bomo morali zadovoljiti s podatki o tem, katere predmete so študentje izbirali. Torej so naše ocene predmetov binarne - 0 (ni izbral predmeta) in 1 (je izbral predmet).

### 2.3.1 Iskanje podobnosti med študenti

Pri prvem načinu priporočanja predmetov bomo iskali med seboj podobne študente. Filtriranje torej temelji na samih študentih - to so naše "stranke" oz. "kupci" (*angl. user-based filtering*). Glavni namen iskanja podobnosti

med študenti je, da nekemu študentu priporočamo predmet, ki ga je pred njim obiskoval njemu podoben študent. Predpostavili bomo torej, da sta si dva študenta podobna takrat, ko izbirata iste predmete in različna takrat, ko vsak izbere druge predmete.

Kot smo omenili v poglavju priprave podatkov 2.1.2, imamo na voljo podatke o izbiri izbirnih predmetov za študijski leti 2010/2011 in 2011/2012. V teh podatkih bomo s pomočjo dveh različnih metod (računanje Jaccard-ovega koeficienta in binarne kosinusne podobnosti) iskali podobnosti med študenti.

Prvo metodo računanja Jaccard-ovega koeficienta (*angl. Jaccard similarity coefficient*) smo uporabili že pri odkrivanju znanj iz besedilnih opisov. Tokrat med seboj delimo število skupnih predmetov študentov A in B s številom vseh predmetov, ki jih obiskujeta študenta A in B. Izračun koeficienta je prikazan v enačbi (2.9).

$$J(A, B) = \frac{\mid A \cap B \mid}{\mid A \cup B \mid} \tag{2.9}$$

Podobnost študentov, ki ne bosta imela nobenega skupnega predmeta, bo enaka 0 ($\mid A \cap B \mid = 0$). V nasprotnem primeru bo podobnost med študenta z identičnim predmetnikom enaka 1 ($\mid A \cap B \mid = \mid A \cup B \mid$).

Čeprav nimamo podrobnejših ocen študentov o predmetih, pa vseeno lahko poskusimo računanje podobnosti s pomočjo kosinusne podobnosti (pri računanju podobnosti med predmeti s pomočjo ključnih oznak predmetov smo uporabili TF-IDF uteži oznak in ne binarnih vrednosti). Našo binarno množico podatkov (0 - ni izbral, 1 - je izbral) lahko tretiramo kot nakupovalno košarico, v kateri je zapisano, katere izdelke je določeni kupec že kupil in katerih še ni. S podobnim problemom so se srečali tudi pri spletni trgovini Amazon.com, kjer so razvili algoritem *Item-to-Item*, ki med izdelki (v našem primeru študijskimi predmeti) računa podobnost s pomočjo modificirane kosinusne podobnosti [8].

Kosinusna podobnost pravzaprav računa kot med vektorjema vrednosti v n-dimenzionalnem prostoru (n je število vrednosti v vektorju). Kosinusna podobnost med študentoma A in B je prikazana v enačbi (2.10)[5].

$$sim(A, B) = cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| \cdot ||\vec{b}||} \tag{2.10}$$

V enačbi (2.10) vektorja $\vec{a}$ in $\vec{b}$ predstavljata binarna vektorja študentov A in B (0 - ni izbral, 1- je izbral). Vektorja vsebujeta toliko binarnih vrednosti,

[5]Cosine similarity.
Dostopno na: `http://en.wikipedia.org/wiki/Cosine_similarity`

kolikor je izbirnih študijskih predmetov.

Izvorno kodo (spisano v programskem jeziku Python) obeh predstavljenih metod za računanje podobnosti pri skupinskem filtriranju najdemo v dodatku D na koncu diplomske naloge (glej izvorno kodo calculateSimilarity_CF.py).

### 2.3.2 Iskanje podobnosti med predmeti

Tako kot smo iskali podobnosti med študenti, pa lahko iščemo tudi podobnosti med predmeti - filtriranje temelji na izdelkih (*angl. item-based filtering*). Ubrali bomo podoben pristop kot pri odkrivanju znanj iz opisov predmetov v poglavju 2.2. Razlika je le v tem, da pri filtriranju ne bomo iskali podobnih predmetov glede na njihove ključne oznake, ampak glede na število skupnih študentov.

Metode, ki smo jih uporabili pri iskanju podobnih študentov, zdaj lahko uporabimo še pri iskanju podobnih predmetov. Tukaj je situacija obrnjena: predmeta sta si med seboj bolj podobna, če imata več skupnih študentov in sta si popolnoma različna, če v preseku nimata nobenega skupnega študenta.

Podobnosti med predmeti bomo torej določali s funkcijama, ki računata podobnost po Jaccard-ovem koeficientu in binarni kosinusni podobnosti. Za primer si lahko pogledamo tabelo 2.3, kjer je za predmet *Elektronsko in mobilno poslovanje* prikazanih pet najbolj podobnih predmetov izračunanih po obeh opisanih metodah.

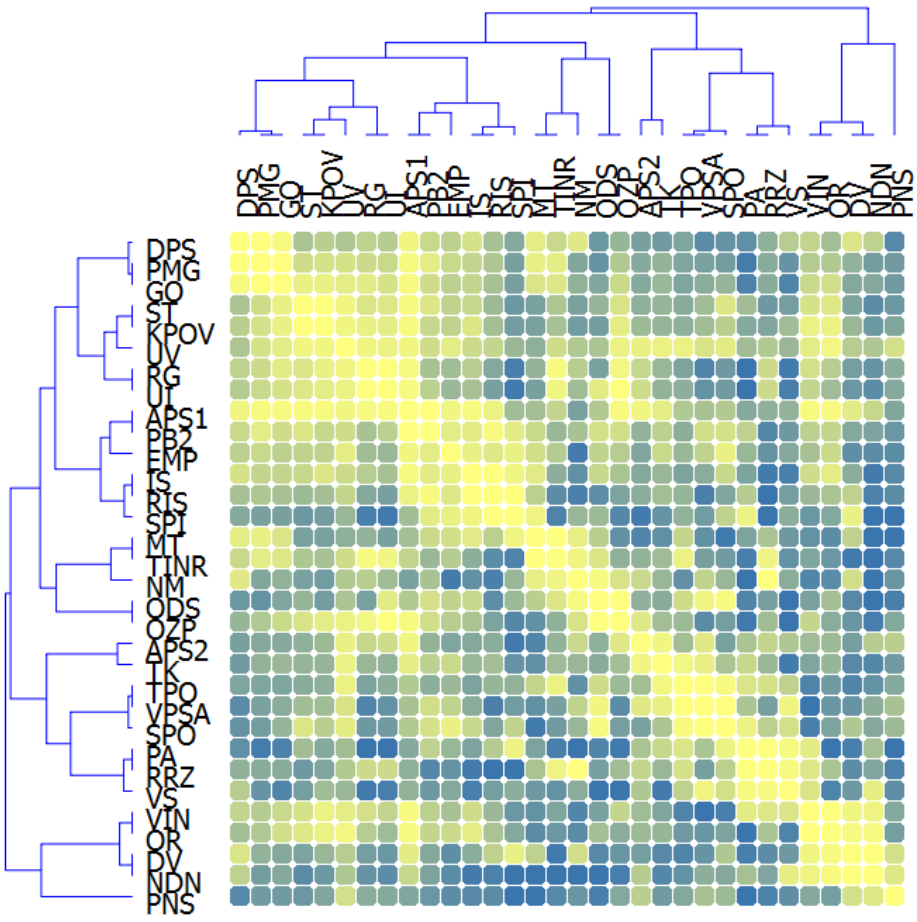
Tabela 2.3: Prikaz petih najbolj podobnih predmetov predmetu *EMP* pri skupinskem filtriranju (izračunano po dveh metodah)

.

| **Jaccard-ov koeficient** | **Binarna kosinusna podobnost** |
|---|---|
| Informacijski sistemi | Algoritmi in podat. strukture 1 |
| Kom. protokoli in omrežna varnost | Informacijski sistemi |
| Algoritmi in podat. strukture 1 | Kom. protokoli in omrežna varnost |
| Podatkovne baze 2 | Podatkovne baze 2 |
| Uporabniški vmesniki | Uporabniški vmesniki |

Prikaz podobnosti med predmeti skupinskega filtriranja vidimo na toplotni karti na sliki 2.5. Tudi tukaj hierarhično razporejanje v skupine predmete razporedi po področjih. Torej študentje v določeni meri izbirajo predmete s sorodnih področij. V to jih nekoliko "prisili" sistem z raznimi omejitvami pri izbiranju predmetov - predpogoji, omejitvami glede ciljnih predmetov ipd.

Na toplotni karti 2.5 pa lahko opazimo tudi, da so nekateri predmeti izbrani zelo pogosto (kot npr. APS1, UV), zato je v njihovih vrsticah/stolpcih več rumene barve. Spet drugi predmeti pa so pri študentih bolj redka izbira (npr. PNS, NM). Take "izjeme" pri izbiranju študijskih predmetov s sistemom odkrivanja znanj iz besedilnih opisov ne bomo morali upoštevati. Na drugi strani pa bo na tovrstne vzorce izbiranja občutljiv sistem skupinskega filtriranja pri iskanju podobnih študentov. Nekoliko izčrpnejši komentar o uspešnosti posameznih sistemov bomo predstavili v naslednjem poglavju 3 vrednotenja priporočilnih sistemov.

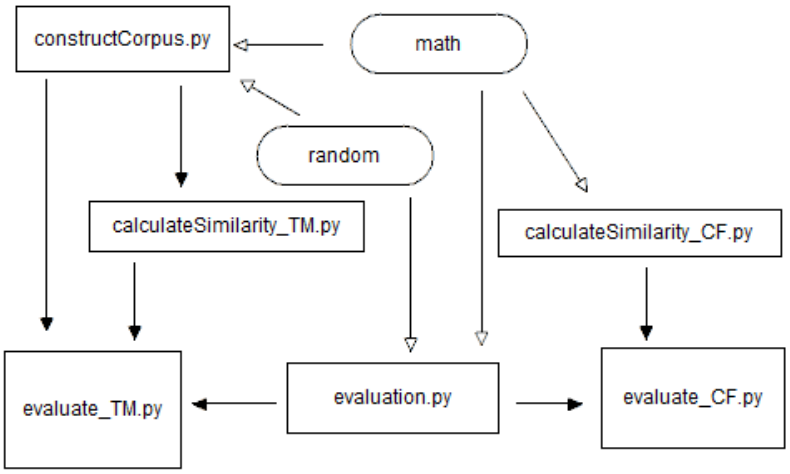



Slika 2.5: Vizualizacija podobnosti med predmeti izračunana pri skupinskem filtriranju (Jaccard-ov koeficient).

## 2.4 Implementacija

Programi za realizacijo obeh sistem za priporočanje in programi za vrednotenje teh sistemov so napisane v programskem jeziku Python. Pri implementaciji smo uporabljali le standardni knjižnici `math` in `random`.

Programsko kodo smo razdelili v smiselne module, katerih diagram je predstavljen na sliki 2.6. Vseh programskih vrstic, skupaj s komentarji, je nekaj več kot 1400. V tabeli 2.4 so na kratko predstavljene funkcionalnosti posameznih modulov.




Slika 2.6: Diagram programskih modulov in Python knjižnic (s puščicami so ponazorjene povezave med moduli)

Tabela 2.4: Na kratko predstavljene funkcionalnosti posameznih programskih modulov, skupaj s podatkom o številu vrstic programske kode posameznega modula (LOC).

| **naziv modula** | **LOC** | **funkcionalnosti modula** |
|---|---|---|
| consturctCorpus.py | 289 | funkcije za branje opisov iz tekstovnih datotek, formiranje korpusa ključnih oznak predmetov (formiranje korpusa tudi pri združenih opisih) |
| calculateSimilarity_TM.py | 168 | funkcije za izračun podobnosti med predmeti po različnih merah pri sistemu odkrivanja znanj iz besedilnih opisov (obe vrsti sistemov - posamezni/združeni opisi) |
| evaluation.py | 235 | branje podatkov iz tekstovnih datotek, izbira testnih predmetov, določanje kandidatov predmetov, določanje ranga pri razporejanju predmetov |
| evaluate_TM.py | 288 | funkcije za izračun povprečnega ranga uvrstitve testnega predmeta pri razporejanju za obe vrsti sistema odkrivanja znanj iz opisov (posamezni/združeni opisi) |
| calculateSimilarity_CF | 165 | funkcije za izračun podobnosti med predmeti/študenti po različnih merah pri sistemu skupinskega filtriranja (obe vrsti sistemov - item/user based) |
| evaluate_CF.py | 273 | funkcije za izračun povprečnega ranga uvrstitve testnega predmeta pri razporejanju za obe vrsti sistema skupinskega filtriranja (item/user based) |

# Poglavje 3

# Vrednotenje uspešnosti priporočilnega sistema

Da bi lahko ocenili, kako dobro so se odrezale posamezne tehnike priporočanja, potrebujemo podatke o tem, kako so predmete študentje izbirali do sedaj. Vrednotenje sistemov bo torej potekalo na resničnih podatkih o izbiri izbirnih predmetov na visokošolskem študiju Fakultete za računalništvo in informatiko v študijskih letih 2010/2011 in 2011/2012. Rezultati nekaterih testiranj bodo v nadaljevanju prikazani v obliki grafikonov. Vsi rezultati testiranj pa so v obliki tabel prikazani tudi pri dodatkih diplomske naloge B in C.

Vrednotenje posameznih metod priporočanja se začne tako, da vsakemu študentu vzamemo po en (ali tudi več predmetov) naključni predmet iz njegovega letnika (bodisi 2. ali 3. letnika) - to je *testni predmet*. Nato poiščemo vse *predmete kandidate*, ki bi jih lahko opazovani študent izbral namesto testnega. Med kandidati je seveda tudi testni predmet. Kandidate rangiramo po ustreznosti glede na ostale predmete, ki jih je izbral opazovani študent. Rangiranje predmetov kandidatov poteka na več načinov glede na izbrano metodo priporočanja (več o tem v naslednjih poglavjih). Rangiranje poteka preprosto: predmetu kandidatu, ki bi ga študentu priporočali kot prvo izbiro, podamo rang 1.0, kandidatu, ki bi ga študentu priporočali kot zadnjo izbiro, pa damo rang 0.0. Ostalim kandidatom ustrezno dodelimo range med 0.0 in 1.0, glede na njihovo ustreznost (od najbolj ustreznega do najmanj ustreznega). Na koncu pogledamo še, kako se je med vsemi kandidati odrezal naš testni predmet. Če smo za najbolj ustreznega med kandidati določili prav testni predmet, torej predmet, ki ga je študent dejansko izbral, bo rang tega predmeta 1.0, sicer pa bo rang manjši od 1.0, morda bo celo enak 0.0. Potek jedra vrednotenja, ki je enak pri vrednotenju vseh sistemov priporočanja, je prikazan na sliki 3.1.

**Algorithm 3.0.1:** EVALUATERECOMMENDATIONSYSTEM(*method*)

```
subjectsList ← GETSUBJECTSLIST()
studentsList ← GETSTUDENTSLIST()

avrRange ← 0.0
corpus ← GETWEIGHTEDCORPUS()

for student ∈ studentsList :
    range ← 0.0
    testSubject ← GETRANDOMTESTSUBJECT(subjectsList[student])
    candidateList ← GETCANDIDATES(student, subjectsList, testSubject)

    range ← CALCULATERANGE(student, subjectsList, testSubject, corpus, method)

    avrRange ← avrRange + range

avrRange ← avrRange/length(studentsList)
return (avrRange)
```

Slika 3.1: Jedro vrednotenja posameznih metod priporočanja.

Pri določanju predmetov kandidatov mora paziti na veliko omejitev:

1. med kandidati ni predmetov, ki jih je študent že izbral,
2. med kandidati ni predmeta, za katerega študent nima izpolnjenih ustreznih pogojev (opravljenih oz. izbranih določenih predmetov),
3. študent v 2. letniku lahko izbira le predmete iz ustreznega semestra (zimskega/poletnega), medtem ko lahko v tretjem letniku izbira predmete iz 2. in 3. letnika (ne glede na semester[1]),
4. študent v 2. letniku ne mora izbrali predmetov iz 3. letnika.

[1]V predstavitvenem zborniku sicer piše, da lahko v 3. letniku izbirajo študentje le predmete iz zimskih semestrov 2. in 3. letnika, vendar podatki kažejo na to, da so študentje v 3. letniku obiskovali tudi predmete iz 2. letnika poletnega semestra.

Preden se lotimo vrednotenja posameznih metod priporočanja, preizkusimo program za testiranje tako, da predmetom kandidatom določamo range povsem naključno (kandidate med seboj pomešamo in jim določimo range od 0.0 do 1.0).

Program za testiranje vrednotenja smo poganjali večkrat in povprečen rang testnih predmetov se vsakič gibal okrog 0.5. To kaže na pravilen potek vrednotenja, saj si lahko naključno določanje rangov predmetom kandidatom predstavljamo kot *ugibanje, kateri predmet bo študent dejansko izbral* - enkrat uganemo, drugič ne.

V nadaljevanju si bomo pogledali, kako uspešno rangiramo predmete kandidate pri različnih metodah priporočanja.

## 3.1 Vrednotenje odkrivanje znanj iz besedilnih opisov

Odkrivanje znanj iz besedilnih opisov (*angl. text mining*) temelji na iskanju podobnosti med besedili, v našem primeru opisi predmetov. Študentu priporočamo predmete, katerih opisi se najbolj skladajo z opisi njegovih že izbranih predmetov. Torej rang kandidatom določamo na podlagi podobnosti med predmeti kandidati in izbranimi predmeti študenta. Podobnost med predmeti oz. opisi pa računamo s pomočjo metod opisanih v poglavju 2.2.5 (Dice-ov in Jaccard-ov koeficient ter kosinusni podobnosti). Potek vrednotenja je prikazan na sliki 3.2.

Pri drugem načinu priporočanja na podlagi odkrivanja znanj iz besedilnih opisov pa opise vseh izbranih predmetov študenta (razen testnega predmeta) združimo. Rang kandidatom določamo na podlagi podobnosti kandidata z združenim opisom. Potek vrednotenja tega drugega načina vidimo na sliki 3.3.

Postopek vrednotenja ponovimo tudi tako, da vsakemu študentu določimo več testnih predmetov in ne le enega, kot je opisano v prejšnjih postopkih. Za testne predmete vzamemo nekaj predmetov iz zadnjega letnika študenta (če je študent v tretjem letniku, vzamemo vse premete iz tretjega letnika, če pa je študent v drugem letniku pa vzamemo tri naključne predmete iz drugega letnika, istega semestra).

---

Potek vrednotenja sistema odkrivanja znanj iz besedilnih opisov (opise predmetov obravnavamo ločeno)

---

1. $\forall$ študent $s_i$ iz množice študentov $S = \{s_i; i \in [1, N]\}$ :

   1.1 izloči testni predmet $p_t$ iz množice izbranih predmetov študenta $s_i$ $P_{s_i} = \{p_1, p_2, ...p_t, ..., p_n\}$; n = število izbranih predmetov

   1.2 določi kandidate predmete glede na omejitve: $C_{s_i} = \{c_1, ..., c_t, ..., c_m\}$; $p_t = c_t$, m = število kandidatov

   1.3 $\forall p_i$ (razen $p_t$) iz množice $P_{s_i}$:

   a) izračunaj podobnosti predmeta $p_i$ z vsemi kandidati iz $C_{s_i}$ [2]: $Sim_{s_i} = \{sim(p_i, c_1), ..., sim(p_i, c_t), ...sim(p_i, c_m)\}$

   b) razporedi podobnosti od največje do najmanjše: $sort(Sim_{s_i}) = \{sim_{max}, ..., sim_{it}, ..., sim_{min}\}$

   c) določi range posameznim podobnostim: $setRange(sort(Sim_{s_i})) = \{r_{sim_{max}} = 1.0, ..., r_{sim_{min}} = 0.0\}$

   d) izloči rang testnega predmeta:

   $$r_{it} = r_{sim_{it}}$$

   1.4 izračunaj povprečen rang iz vseh rangov $r_{it}$, ki jih je testni predmet dosegel pri izračunu podobnosti s predmeti iz $P_{s_i}$ :

   $$r_{avr_{s_i}} = \frac{\sum_{j=1}^{n-1}(r_{ij})}{n-1}$$

2. izračunaj povprečen rang vseh povprečnih rangov $r_{avr_{s_i}}$ vseh študentov iz množice študentov $S$:

   $$R_{avr} = \frac{\sum_{i=1}^{N}(r_{avr_{s_i}})}{N}$$

---

Slika 3.2: Vrednotenje sistema odkrivanja znanj iz besedilnih opisov predmetov, pri katerem te opise obravnavamo ločeno.

---

Potek vrednotenja sistema odkrivanja znanj iz besedilnih opisov (opise predmetov obravnavamo skupaj)

---

1. $\forall$ študent $s_i$ iz množice študentov $S = \{s_i; i \in [1, N]\}$ :
   - 1.1 izloči testni predmet $p_t$ iz množice izbranih predmetov študenta $s_i$ $P_{s_i} = \{p_1, p_2, ...p_t, ..., p_n\}$; n = število izbranih predmetov
   - 1.2 določi kandidate predmete, glede na omejitve: $C_{s_i} = \{c_1, ..., c_t, ..., c_m\}$; $p_t = c_t$, m = število kandidatov
   - 1.3 združi opise predmetov ($\zeta$) iz množice $P_{s_i}$ (vse razen $p_t$) in na novo izračunaj uteži TF-IDF za vse združene ključne besede
   - 1.4 izračunaj podobnost med kandidati iz $C_{s_i}$ in združenim opisom $\zeta$ [3]: $Sim_{s_i} = \{sim(\zeta, c_1), ..., sim(\zeta, c_t), ..., sim(\zeta, c_m)\}$
   - 1.5 razporedi podobnosti od največje do najmanjše: $sort(Sim_{s_i}) = \{sim_{max}, ..., sim_{\zeta t}, ..., sim_{min}\}$
   - 1.6 določi range posameznim podobnostim: $setRange(sort(Sim_{s_i})) = \{r_{sim_{max}} = 1.0, ..., r_{sim_{min}} = 0.0\}$
   - 1.7 izloči rang testnega predmeta:

$$r_{s_i} = r_{sim_{\zeta t}}$$

2. izračunaj povprečen rang testnih predmetov $r_{s_i}$ vseh študentov iz množice študentov $S$:

$$R_{avr} = \frac{\sum_{i=1}^{N}(r_{s_i})}{N}$$

---

Slika 3.3: Vrednotenje sistema odkrivanja znanj iz besedilnih opisov predmetov, pri katerem te opise obravnavamo ločeno.

## 3.2 Vrednotenje skupinskega filtriranja

Pri vrednotenju skupinskega filtriranja (*angl. collaborative filtering*) bomo podatke, na katerih testiramo, uporabili tudi pri računanju podobnosti med študenti in med predmeti.

Glavna ideja računanja podobnosti med študenti (*angl. user-based*) je, da študentu priporočamo predmete, ki so jih v preteklosti izbrali njemu karseda podobni kolegi. Naš pristop je preprost. Poiščemo nekaj najbolj podobnih študentov z metodami opisanimi v poglavju 2.3.1. Potem štejemo, kako pogosto se predmeti kandidati pojavijo pri teh podobnih študentih. Potek vrednotenja skupinskega filtriranja na osnovi podobnosti študentov je prikazan na sliki 3.4.

Drugi način priporočanja na podlagi skupinskega filtriranja pa temelji na podlagi izračunavanja podobnosti med predmeti (*angl. item-based*), katere ne računamo na podlagi opisov predmetov, ampak na podlagi skupnih študentov, ki jih imata dva opazovana predmeta. Dva predmeta sta si bolj podobna, čim več skupnih študentov imata. Potek vrednotenja skupinskega filtriranja na osnovi podobnosti predmetov lahko vidimo na sliki 3.5.

Tudi pri skupinskem filtriranju poskusimo vrednotenje z več testnimi predmeti pri vsakem študentu. Testne predmete določamo na isti način kot pri odkrivanju znanj iz besedilnih opisov.

V naslednjem poglavju bomo predstavili tudi rezultate, ki jih dajejo posamezni postopki vrednotenja.

---

Potek vrednotenja sistema skupinskega filtriranja
(iščemo podobne študente)

---

1. $\forall$ študent $s_i$ iz množice študentov $S = \{s_i; i \in [1, N]\}$ :

   1.1 izloči testni predmet $p_t$ iz množice izbranih predmetov študenta $s_i$
   $P_{s_i} = \{p_1, p_2, ...p_t, ..., p_n\}$; n = število izbranih predmetov

   1.2 določi kandidate predmete, glede na omejitve:
   $C_{s_i} = \{c_1, ..., c_t, ..., c_m\}$; $p_t = c_t$, m = število kandidatov

   1.3 poišči najbolj podobne študente iz $S$ študentu $s_i$ [4]:
   $\Omega_{s_i} = \{\omega_1, ...\omega_k\}$; k = število podobnih študentov

   1.4 $\forall c_j$ iz množice $C_{s_i}$ preštej, kolikokrat se pojavi pri študentih iz množice podobnih študentov $\Omega_{s_i}$:
   $F_{\Omega_{s_i}} = \{f_1, ..., f_m\}$

   1.5 razporedi frekvence pojavljanja predmetov iz $F_{\Omega_{s_i}}$ po velikosti od tistega z največjim $f_j$ do tistega z najmanjšim $f_j$ :
   $sort(F_{\Omega_{s_i}}) = \{f_{max}, ..., f_t, ..., f_{min}\}$

   1.6 določi range posameznim frekvencam $f_j$:
   $setRange(sort(F_{\Omega_{s_i}})) = \{r_{f_{max}} = 1.0, ..., r_{f_{min}} = 0.0\}$

   1.7 izloči rang testnega predmeta:

$$r_{s_i} = r_{f_t}$$

2. izračunaj povprečen rang testnih predmetov $r_{s_i}$ vseh študentov iz množice študentov $S$:

$$R_{avr} = \frac{\sum_{i=1}^{N}(r_{s_i})}{N}$$

---

Slika 3.4: Vrednotenje sistema skupinskega filtriranja, pri katerem iščemo podobne študente.

---

POTEK VREDNOTENJA SISTEMA SKUPINSKEGA FILTRIRANJA
(iščemo podobne predmete)

---

1. $\forall$ študent $s_i$ iz množice študentov $S = \{s_i; i \in [1, N]\}$ :

   1.1 izloči testni predmet $p_t$ iz množice izbranih predmetov študenta $s_i$
   $P_{s_i} = \{p_1, p_2, ...p_t, ..., p_n\}$; n = število izbranih predmetov

   1.2 določi kandidate predmete, glede na omejitve:
   $C_{s_i} = \{c_1, ..., c_t, ..., c_m\}$; $p_t = c_t$, m = število kandidatov

   1.3 $\forall p_i$ (razen $p_t$) iz množice $P_{s_i}$:

   a) izračunaj podobnosti predmetov iz $P_{s_i}$ z vsemi kandidati iz $C_{s_i}$ [5]:
   $Sim_{s_i} = \{sim(p_i, c_1), ..., sim(p_i, c_t), ...sim(p_i, c_m)\}$

   b) razporedi podobnosti od največje do najmanjše:
   $sort(Sim_{s_i}) = \{sim_{max}, ..., sim_{it}, ..., sim_{min}\}$

   c) določi range posameznim podobnostim:
   $setRange(sort(Sim_{s_i})) = \{r_{sim_{max}} = 1.0, ..., r_{sim_{min}} = 0.0\}$

   d) izloči rang testnega predmeta kandidata:

   $$r_{it} = r_{sim_{it}}$$

   1.4 izračunaj povprečen rang iz vseh rangov $r_{it}$, ki jih je testni predmet dosegel pri izračunu podobnosti s predmeti iz $P_{s_i}$:

   $$r_{avr_{s_i}} = \frac{\sum_{j=1}^{n-1}(r_{ij})}{n-1}$$

2. izračunaj povprečen rang vseh povprečnih rangov $r_{avr_{s_i}}$ vseh študentov iz množice študentov $S$:

   $$R_{avr} = \frac{\sum_{i=1}^{N}(r_{avr_{s_i}})}{N}$$

---

Slika 3.5: Vrednotenje sistema skupinskega filtriranja, pri katerem iščemo podobne študijske predmete.

## 3.3 Rezultati vrednotenja in primerjava sistemov za priporočanje

Posamezne sisteme za priporočanje bomo v nadaljevanju testirali s postopki vrednotenja, ki smo jih predstavili v poglavjih 3.1 in 3.2. Rezultate vrednotenja implementiranih postopkov priporočanja bomo prikazali na več grafikonih, katere bomo tudi podrobno komentirali. Pregled vseh rezultatov teh testiranj nam bo pomagal pri odločanju o morebitnem nadaljnjem razvoju sistema za priporočanje.

### 3.3.1 Rezultati vrednotenja odkrivanja znanj iz besedilnih opisov

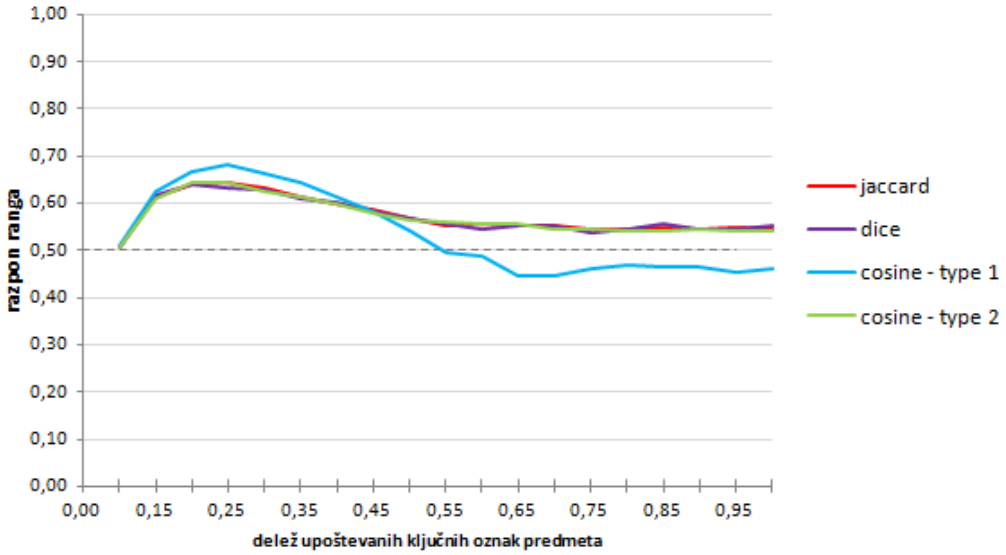
Najprej bomo vrednotili odkrivanje znanj iz opisov predmetov. Na posameznih grafih je prikazano, kako se spreminja povprečen rang testnega predmeta v odvisnosti od deleža ključnih oznak, ki jih zajamemo pri posameznem predmetu. Kot vemo, so oznake za posamezen predmet v gnezdenem slovarju razporejene od tiste z najvišjo do oznake z najnižjo TF-IDF utežjo. Oznake z najnižjo težo so torej relativno nepomembne in jih pri izračunavanju podobnosti predmetov lahko tudi ignoriramo.

Vsakega od postopkov vrednotenja poganjamo večkrat. Vsakič so rezultati nekoliko drugačni, saj naključno izbiramo testni predmet. V grafih so prikazani povprečni rezultati večkratnega testiranja.

Na grafih 3.6 in 3.7 so prikazani podatki vrednotenja vseh štirih metod računanja podobnosti med predmeti:

- Dice-ov koeficient (vijoličasta barva)
- Jaccard-ov koeficient (rdeča barva)
- kosinusna podobnost (modra barva)
- poenostavljena kosinusna podobnost (zelena barva)

Na sliki 3.6 vidimo rezultate prvih testiranj. Vrednotimo odkrivanja znanj iz besedilih opisov, kjer vsak opis predmeta obravnavamo ločeno. Graf na sliki 3.6 prikazuje, kako se spreminja povprečen rang testnih predmetov ob spreminjanju deleža upoštevanih ključnih oznak (upoštevamo vedno tiste oznake, ki imajo najvišje uteži). Na grafu lahko vidimo, da je najbolje vzeti približno le 20 do 25 odstotkov vseh ključnih oznak pri posameznih predmetih, saj se testni predmeti pri tem deležu v povprečju najvišje rangirajo (do skoraj 0.7).

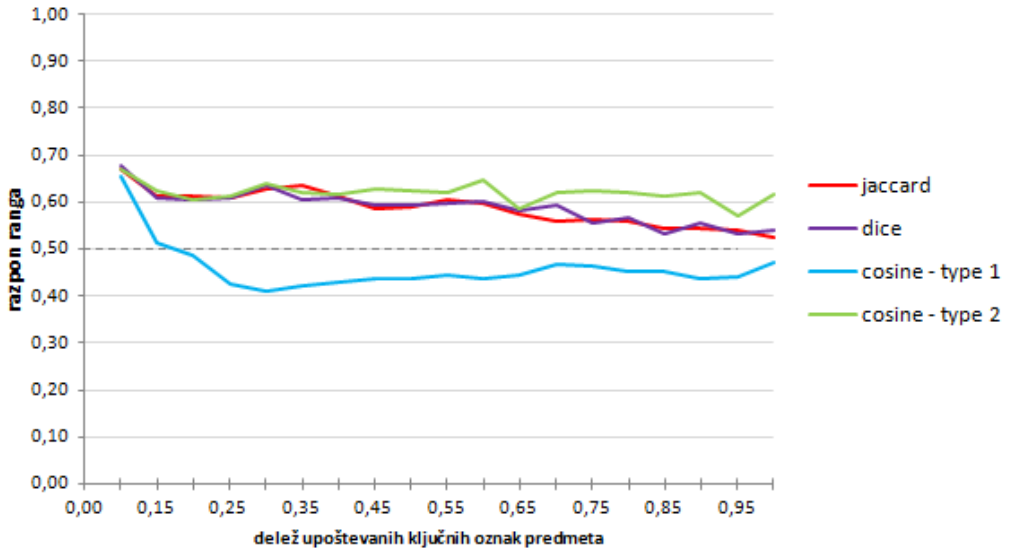



Slika 3.6: Graf povprečnega ranga odkrivanja znanj iz besedilnih opisov ob spreminjanju deleža upoštevanih ključnih oznak predmetov (en testni predmet na študenta, opisi predmetov obravnavani ločeno).

Graf prikazan na sliki 3.6 nam pokaže, da se kot najboljša metoda za priporočanje predmetov iz sorodnih področij izkaže računanje kosinusne podobnosti med predmeti (ob zajetju četrtine oziroma petine vseh oznak predmetov). Pri opisovanju metod za računanje podobnosti v poglavju 2.2.5 se nam je klasična kosinusna podobnost zdela na prvi pogled najmanj "očitna" pri svojih rezultatih, vendar v praksi deluje najbolje. Je pa res, da se ob zajemu dobre polovice oznak, kosinusna metoda izkaže za slabši postopek od ugibanja (rang pod pragom 0.5). Medtem se ostale metode, ki so po uspehu med seboj skoraj identične, ves čas držijo nad rangom 0.5.

Zelo podobni so rezultati testiranj odkrivanja znanj iz besedilnih opisov, pri katerem vsakem študentu vzamemo več testnih predmetov (za najboljšo se izkaže klasična kosinusna metoda, pri 15 do 20 odstotkih upoštevanih ključnih oznak se rang dvigne tudi malo nad 0.7).

Pri testiranju odkrivanja znanj iz besedilnih opisov, kjer opise predmetov vsakega študenta združimo v en skupen opis, dobimo nekoliko drugačne rezultate. Na sliki 3.7 vidimo, da prav vse metode najboljše rangirajo (rang blizu 0.7) pri zajemu le desetine najpomembnejših ključnih oznak. To je precej razumljivo, saj se količina oznak pri združenem opisu razširi za faktor števila združenih predmetov. Ključnih besed je tako nekajkrat več, kot pri običajnem postopku, kjer opise obravnavamo povsem ločeno.



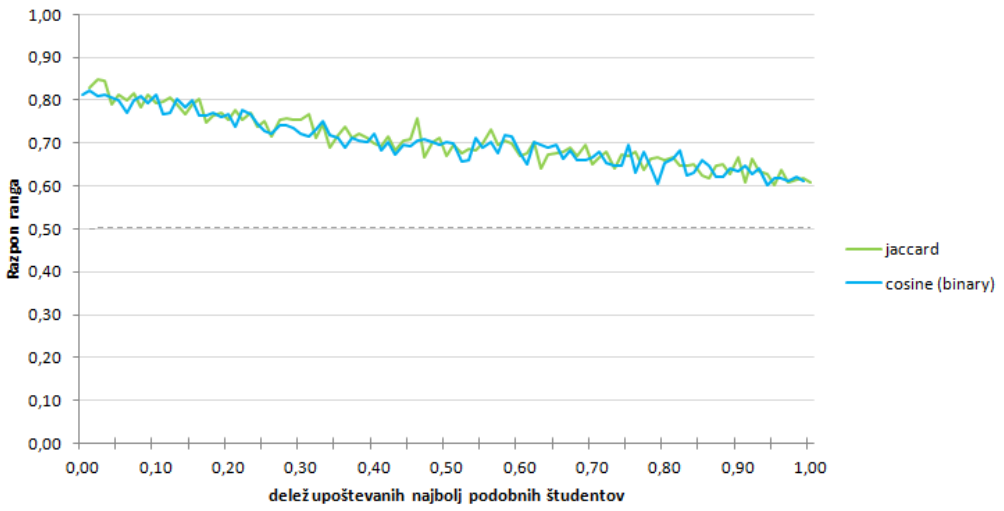

Slika 3.7: Graf povprečnega ranga odkrivanja znanj iz besedilnih opisov ob spreminjanju deleža upoštevanih ključnih oznak predmetov (en testni predmet na študenta, združeni opisi predmetov enega študenta).

Dosedanji rezultati kažejo, da samo odkrivanje znanj iz besedilnih opisov ni najboljša metoda za priporočanje izbirnih predmetov. Najvišji povprečni rangi so se gibali okoli 0.67 do 0.69, kar kaže na nekaj manj kot 70-odstotno uspešnost priporočanja. K slabšemu uspehu bi lahko pripomogla neprava izbira metod za računanje podobnosti med predmeti, ali pa ne dovolj izčrpni opisi predmetov, iz katerih smo pridobivali ključne oznake predmetov. Ker pa so se nam zdele metode na prvi pogled dokaj uspešne (za podobne predmete so metode označile predmete iz sorodnih področij), bi lahko rekli tudi, da študentje ne izbirajo vedno predmetov iz sorodnih področij in jim tovrstna priporočila ne ustrezajo.

### 3.3.2 Rezultati vrednotenja skupinskega filtriranja

Pri vrednotenju skupinskega filtriranja testiramo dve metodi računanja podobnosti med študenti oz. predmeti - Jaccard-ov koeficient (na grafu prikazano z zeleno) in kosinusna podobnost (na grafu prikazano z modro). V naslednjih grafikonih bo prikazano spreminjanje povprečnega ranga v odvisnosti od upoštevanega deleža najbolj podobnih študentov, ki jih vključimo v računanje podobnosti in priporočanje.

Na sliki 3.8 je prikazan prvi graf vrednotenja skupinskega filtriranja (računanje podobnosti med študenti). Sistem za priporočanje se najbolj obnese z Jaccard-ovim koeficientom (rang sega do 0.85) pri zajemu zgolj enega ali dva odstotka najbolj podobnih študentov. Ker imamo na razpolago nekaj manj kot 200 študentov, to pomeni, da je dovolj, da študentu poiščemo enega ali dva najbolj podobna študenta in že lahko napovemo, kdaj bo opazovani študent izbral. Nekoliko slabše pa vseeno dobre rezultate vrednotenja priporočanja dobimo, če vsakemu študentu določimo več testnih predmetov.




Slika 3.8: Graf povprečnega ranga vrednotenja skupinskega filtriranja študentov ob spreminjanju deleža upoštevanih podobnih študentov (en testni predmet na študenta).

Pri vrednotenju drugega načina skupinskega filtriranja računamo podobnosti med študijskimi predmeti. Pri računanju podobnosti enega predmeta z ostalimi moramo seveda zajeti prav vse predmete (in ne le deleža predmetov, kot smo to storili pri študentih, kjer smo zajemali le delček najbolj podobnih študentov). Pri večkratnem poskusu vrednotenja priporočanja na podlagi podobnosti med predmeti, kjer za test vzamemo po en predmet pri vsakem študentu, se rang giba med 0.65 in slabih 0.7 (velja za obe metodi - Jaccard-ov koeficient in kosinusna podobnost). Pri vrednotenju sistema z več testnimi predmeti pri vsakem študentu so rezultati zopet nekoliko slabši.

Na podlagi teh rezultatov vrednotenja skupinskega filtriranja lahko zaključimo, da se filtriranje podobnih študentov glede na izbiro predmetov (user-based) obnese veliko bolje kot filtriranje podobnih predmetov (item-based).

# Poglavje 4

# Sklepne ugotovitve

Glede na rezultate vrednotenja sistemov bi lahko rekli, da je najboljši sistem za priporočanje skupinsko filtriranje, pri katerem iščemo podobne študente. Morda se sliši nekoliko neverjetno, toda izbiro predmetov nekega študenta lahko precej natančno napovemo že tako, da poiščemo študente, ki so v preteklosti izbirali podobne predmete kot opazovani študent. Rezultati kažejo, da se študentje ne odločajo vedno za predmete iz istih oz. sorodnih področij (nekateri predmeti so izbrani zelo pogosto, drugi manj pogosto), zato se sistem odkrivanja znanj iz opisov ni odrezal najbolje. Očitno študentje predmete izbirajo tipično raznoliko, kar pomeni, da se pri njihovih preteklih izbirah vseeno lahko najdejo določeni (ponavljajoči se) vzorci, ki nam pomagajo pri napovedovanju, ali bo neki študent v prihodnje izbral določeni predmet ali ne.

Kljub tej ugotovitvi pa se moramo zavedati, da se pri razvoju priporočilnega sistema srečujemo z že omenjenim pojavom hladnega zagona. Ko v sistem dobimo nekega novega študenta, ki še ni izbiral nobenih predmetov, skupinsko filtriranje ne deluje, saj novincu ne moramo določiti podobnih študentov. V takem primeru lahko nekaj podatkov o študentu pridobimo bodisi eksplicitno (z vprašalniki) ali implicitno (splošni podatki o študentu, recimo srednja šola, predznanja, ipd.) in uporabimo drugo vrsto sistema za priporočanje. V takem primeru bi lahko npr. sestavili poseben vprašalnik o interesih, ki bi ga študent "zapolnil s ključnimi besedami". Na podlagi teh besed bi lahko s pomočjo odkrivanja znanj iz besedilnih opisov izračunali, kateri predmeti najbolj ustrezajo vprašanemu.

Glavna ugotovitev diplomske naloge je torej, da je pri implementaciji priporočilnega sistema za izbiro izbirnih študijskih predmetov najbolje razvijati sistem, ki temelji na skupinskem filtriranju, predvsem zaradi tipično raznolikega izbiranja predmetov s strani študentov. Dodatno pa je potrebno pou-

dariti, da nam tudi sistem odkrivanja znanj iz besedilnih opisov pride prav predvsem zaradi izpostavljenega problema hladnega zagona.

## 4.1 Možne izboljšave sistema

Idej za izboljšavo sistema je kar nekaj. Če bi tovrsten sistem realizirali v praksi, bi bilo najprej dobro poskrbeti za izčrpnejše podatke. Verjetno bi bilo odkrivanje znanj iz besedilnih opisov še nekoliko boljše, če bi imeli natančnejše opise predmetov. Sistem bi tako našel še več pomembnih ključnih oznak pri vsakem predmetu in priporočila bi bila lahko bolj natančna. Tudi na področju skupinskega filtriranja bi prišlo prav še več podatkov o preteklih izbirah predmetov.

Sistem, ki smo ga razvili, je bolj prototipen - namenjen raziskavi. Če bi se priporočilni sistem začel uporabljati na več fakultetah hkrati ali pa celo na ravni celotne univerze, bi bilo potrebno razmisliti o konkretnih bazah podatkov študijskih predmetov in študentov, ki so te predmete izbirali. Prav tako bi bilo potrebno sistem "pohitriti", da bi priporočila delovala v realnem času in sistem seveda tudi sproti vzdrževati in nadgrajevati.

## 4.2 Ideje za nadaljnje delo

Sistem za priporočanje pri izbiri študijskih predmetov ne bi bil koristen samo na posameznih fakultetah, ampak na ravni celotne univerze. Kot smo že uvodoma namignili, bi lahko naša univerza poskrbela tudi na nekakšen obsežnejši spletni pregled vseh študijskih programov in predmetov, ki jih njene fakultete in akademije ponujajo. Tako bi imeli študentje, ki lahko po novem izbirajo predmete na več fakultetah, manj težav pri iskanju potrebnih informacij o možnih smereh svojega študija. Pri odločanju pa bi jim zagotovo koristil tudi interaktivni pregledovalnik in avtomatski svetovalec, ki bi jih usmerjal s svojimi priporočili. Skupni portal vseh fakultet, kjer bi bili registrirani vsi študentje univerze (redni, izredni, tuji, domači,...) pa bi služil tudi kot učinkovit "zbiratelj" prepotrebnih informacij za realizacijo in nadgrajevanje tovrstnih priporočilnih sistemov.

Poleg same integracije priporočilnega sistema v spletni študijski portal, pa bi nadaljnje delo na sistemu posvetili predvsem dodatnim obdelavam podatkov in iskanjem zakonitosti v podatkih. Z iskanjem vzorcev izbiranja predmetov bi še lažje razumeli odločitve študentov, kar bi seveda prispevalo k razvoju in izboljšavam tovrstnih priporočilnih sistemov v prihodnosti.

# Dodatek A

# Seznam študijskih predmetov

Tabela A.1: Preglednica polnih nazivov študijskih predmetov z oznakami.

| **Oznaka** | **Polni naziv predmeta** |
|---|---|
| APS1 | Algoritmi in podatkovne strukture 1 |
| APS2 | Algoritmi in podatkovne strukture 2 |
| DV | Digitalna vezja |
| DPS | Digitalno procesiranje signalov |
| EMP | Elektronsko in mobilno poslovanje |
| GO | Grafično oblikovanje |
| IS | Informacijski sistemi |
| KPOV | Komunikacijski protokoli in omrežna varnost |
| MT | Multimedijske tehnologije |
| NDN | Načrtovanje digitalnih naprav |
| NM | Numerične metode |
| OZP | Odkrivanje zakonitosti iz podatkov |
| ODS | Odločitveni sistemi |
| OR | Organizacija računalnikov |
| PB2 | Podatkovne baze 2 |
| PNS | Prevajalniki in navidezni stroji |
| PA | Procesna avtomatika |
| PMG | Produkcija multimedijskih gradiv |
| RG | Računalniška grafika |
| RIS | Razvoj informacijskih sistemov |
| RRZ | Robotika in računalniško zaznavanje |
| SPO | Sistemska programska oprema |
| ST | Spletne tehnologije |
| SPI | Strateško planiranje informatike |
| TINR | Tehnologija iger in navidezna resničnost |
| TPO | Tehnologija programske opreme |
| TK | Testiranje in kakovost |
| UI | Umetna inteligenca |
| UV | Uporabniški vmesniki |
| VS | Vgrajeni sistemi |
| VIN | Vhodno-izhodne naprave |
| VPSA | Vzporedni in porazdeljeni sistemi in algoritmi |

# Dodatek B

# Rezultati vrednotenja odkrivanja znanj iz besedilnih opisov

V naslednjih preglednicah so predstavljeni rezultati testiranj priporočilnega sistema, ki temelji na odkrivanju znanj iz besedilnih opisov študijskih predmetov. V tabelah so predstavljeni povprečni rezultati večkratnega testiranja sistemov. Sisteme smo testirali z različnimi algoritmi računanja podobnosti med predmeti – Jaccard-ov koeficient podobnosti, Dice-ov koeficient podobnosti, kosinusna podobnost in poenostavljena kosinusna podobnost. Višji rezultat (bližje 1.0) pomeni boljše napovedi o izbirah študijskih predmetov s strani študentov. Sistemi so testirani na resničnih podatkih o izbiranju izbirnih predmetov visokošolskih študentov Fakultete za računalništvo in informatiko v Ljubljani v študijskih letih 2010/2011 in 2011/2012. V tabelah B.1 in B.2 opise predmetov pri računanju podobnosti obravnavamo ločeno, v tabelah B.3 in B.4 pa opise izbirnih predmetov opazovanega študenta obravnavamo skupaj.

Tabela B.1: Povprečen rang vrednotenja odkrivanja znanj iz besedilnih opisov ob spremembah deleža upoštevanih oznak predmetov - vsakemu študentu je določen po en testni predmet, opisi predmetov obravnavani ločeno.

| **delež oznak** | **Jaccard** | **Dice** | **cosine 1** | **cosine 2** |
|---|---|---|---|---|
| 0,10 | 0,506 | 0,507 | 0,510 | 0,506 |
| 0,15 | 0,612 | 0,616 | 0,623 | 0,611 |
| 0,20 | 0,642 | 0,639 | 0,665 | 0,643 |
| 0,25 | 0,643 | 0,633 | 0,680 | 0,643 |
| 0,30 | 0,631 | 0,629 | 0,665 | 0,626 |
| 0,35 | 0,615 | 0,609 | 0,644 | 0,613 |
| 0,40 | 0,600 | 0,600 | 0,614 | 0,598 |
| 0,45 | 0,586 | 0,584 | 0,584 | 0,578 |
| 0,50 | 0,566 | 0,566 | 0,539 | 0,564 |
| 0,55 | 0,554 | 0,557 | 0,496 | 0,560 |
| 0,60 | 0,555 | 0,545 | 0,486 | 0,556 |
| 0,65 | 0,551 | 0,551 | 0,445 | 0,557 |
| 0,70 | 0,553 | 0,554 | 0,448 | 0,544 |
| 0,75 | 0,546 | 0,538 | 0,460 | 0,544 |
| 0,80 | 0,547 | 0,547 | 0,468 | 0,543 |
| 0,85 | 0,549 | 0,556 | 0,466 | 0,542 |
| 0,90 | 0,544 | 0,544 | 0,465 | 0,543 |
| 0,95 | 0,547 | 0,546 | 0,452 | 0,539 |
| 1,00 | 0,544 | 0,551 | 0,462 | 0,543 |

Tabela B.2: Povprečen rang vrednotenja odkrivanja znanj iz besedilnih opisov ob spremembah deleža upoštevanih oznak predmetov - vsakemu študentu je določeno po več testnih predmetov, opisi predmetov obravnavani ločeno.

| **delež oznak** | **Jaccard** | **Dice** | **cosine 1** | **cosine 2** |
|---|---|---|---|---|
| 0,10 | 0,644 | 0,627 | 0,628 | 0,635 |
| 0,15 | 0,706 | 0,712 | 0,723 | 0,708 |
| 0,20 | 0,684 | 0,674 | 0,706 | 0,676 |
| 0,25 | 0,647 | 0,650 | 0,686 | 0,648 |
| 0,30 | 0,625 | 0,622 | 0,663 | 0,621 |
| 0,35 | 0,607 | 0,602 | 0,645 | 0,601 |
| 0,40 | 0,595 | 0,583 | 0,605 | 0,583 |
| 0,45 | 0,571 | 0,565 | 0,575 | 0,561 |
| 0,50 | 0,548 | 0,556 | 0,535 | 0,545 |
| 0,55 | 0,545 | 0,544 | 0,501 | 0,544 |
| 0,60 | 0,540 | 0,540 | 0,499 | 0,536 |
| 0,65 | 0,540 | 0,541 | 0,459 | 0,543 |
| 0,70 | 0,540 | 0,544 | 0,466 | 0,538 |
| 0,75 | 0,535 | 0,534 | 0,474 | 0,536 |
| 0,80 | 0,532 | 0,530 | 0,471 | 0,536 |
| 0,85 | 0,538 | 0,537 | 0,477 | 0,534 |
| 0,90 | 0,539 | 0,531 | 0,476 | 0,530 |
| 0,95 | 0,526 | 0,529 | 0,474 | 0,530 |
| 1,00 | 0,533 | 0,535 | 0,477 | 0,536 |

Tabela B.3: Povprečen rang vrednotenja odkrivanja znanj iz besedilnih opisov ob spremembah deleža upoštevanih oznak predmetov - vsakemu študentu je določen po en testni predmet, opisi predmetov obravnavani skupaj.

| **delež oznak** | **Jaccard** | **Dice** | **cosine 1** | **cosine 2** |
|---|---|---|---|---|
| 0,10 | 0,671 | 0,677 | 0,657 | 0,669 |
| 0,15 | 0,612 | 0,607 | 0,515 | 0,624 |
| 0,20 | 0,611 | 0,604 | 0,488 | 0,606 |
| 0,25 | 0,610 | 0,609 | 0,424 | 0,613 |
| 0,30 | 0,627 | 0,635 | 0,409 | 0,641 |
| 0,35 | 0,637 | 0,607 | 0,421 | 0,622 |
| 0,40 | 0,611 | 0,607 | 0,429 | 0,618 |
| 0,45 | 0,587 | 0,593 | 0,438 | 0,627 |
| 0,50 | 0,588 | 0,593 | 0,435 | 0,623 |
| 0,55 | 0,605 | 0,599 | 0,446 | 0,619 |
| 0,60 | 0,597 | 0,600 | 0,436 | 0,647 |
| 0,65 | 0,576 | 0,582 | 0,446 | 0,588 |
| 0,70 | 0,558 | 0,594 | 0,468 | 0,621 |
| 0,75 | 0,564 | 0,555 | 0,463 | 0,626 |
| 0,80 | 0,558 | 0,568 | 0,451 | 0,619 |
| 0,85 | 0,546 | 0,531 | 0,454 | 0,612 |
| 0,90 | 0,544 | 0,556 | 0,438 | 0,621 |
| 0,95 | 0,542 | 0,531 | 0,440 | 0,570 |
| 1,00 | 0,526 | 0,541 | 0,472 | 0,615 |

Tabela B.4: Povprečen rang vrednotenja odkrivanja znanj iz besedilnih opisov ob spremembah deleža upoštevanih oznak predmetov - vsakemu študentu je določeno po več testnih predmetov, opisi predmetov obravnavani skupaj.

| **delež oznak** | **Jaccard** | **Dice** | **cosine 1** | **cosine 2** |
|---|---|---|---|---|
| 0,10 | 0,685 | 0,697 | 0,675 | 0,688 |
| 0,15 | 0,596 | 0,590 | 0,574 | 0,603 |
| 0,20 | 0,576 | 0,571 | 0,527 | 0,586 |
| 0,25 | 0,572 | 0,566 | 0,483 | 0,570 |
| 0,30 | 0,572 | 0,572 | 0,464 | 0,571 |
| 0,35 | 0,582 | 0,567 | 0,458 | 0,562 |
| 0,40 | 0,563 | 0,577 | 0,464 | 0,556 |
| 0,45 | 0,564 | 0,571 | 0,453 | 0,567 |
| 0,50 | 0,553 | 0,556 | 0,446 | 0,566 |
| 0,55 | 0,577 | 0,576 | 0,452 | 0,572 |
| 0,60 | 0,568 | 0,557 | 0,442 | 0,565 |
| 0,65 | 0,561 | 0,560 | 0,446 | 0,566 |
| 0,70 | 0,560 | 0,558 | 0,449 | 0,583 |
| 0,75 | 0,546 | 0,544 | 0,449 | 0,562 |
| 0,80 | 0,545 | 0,528 | 0,436 | 0,561 |
| 0,85 | 0,533 | 0,538 | 0,448 | 0,573 |
| 0,90 | 0,519 | 0,535 | 0,458 | 0,548 |
| 0,95 | 0,531 | 0,532 | 0,457 | 0,557 |
| 1,00 | 0,522 | 0,529 | 0,472 | 0,561 |

# Dodatek C

# Rezultati vrednotenja skupinskega filtriranja

V dodatku C so predstavljeni rezultati testiranj priporočilnega sistema, ki temelji na skupinskem filtriranju. V tabelah so predstavljeni povprečni rezultati večkratnega testiranja sistemov. Sisteme smo testirali z dvema različnima algoritmoma računanja podobnosti predmetov oz. študentov – Jaccard-ov koeficient podobnosti in binarna kosinusna podobnost. Višji rezultat (bližje 1.0) pomeni boljše napovedi o izbirah študijskih predmetov s strani študentov. Sistemi so testirani na resničnih podatkih o izbiranju izbirnih predmetov visokošolskih študentov Fakultete za računalništvo in informatiko v Ljubljani v študijskih letih 2010/2011 in 2011/2012. V tabelah C.1 in C.2 so prikazani rezultati vrednotenja, pri katerem iščemo podobne študente, v tabelah C.3 in C.4 pa iščemo med seboj podobne predmete.

Tabela C.1: Povprečen rang vrednotenja skupinskega filtriranja ob spremembah deleža upoštevanih podobnih študentov - vsakemu študentu je določen po en testni predmet, iščemo podobnosti med študenti.

| **delež študentov** | **Jaccard** | **binary cosine** |
|---|---|---|
| 0,02 | 0,858 | 0,824 |
| 0,04 | 0,808 | 0,813 |
| 0,06 | 0,800 | 0,800 |
| 0,08 | 0,785 | 0,799 |
| 0,10 | 0,794 | 0,795 |
| 0,12 | 0,805 | 0,769 |
| 0,14 | 0,766 | 0,804 |
| 0,16 | 0,804 | 0,799 |
| 0,18 | 0,765 | 0,765 |
| 0,20 | 0,755 | 0,760 |
| 0,22 | 0,754 | 0,738 |
| 0,24 | 0,737 | 0,767 |
| 0,26 | 0,715 | 0,730 |
| 0,28 | 0,758 | 0,741 |
| 0,30 | 0,753 | 0,735 |
| 0,32 | 0,712 | 0,717 |
| 0,34 | 0,688 | 0,750 |
| 0,36 | 0,737 | 0,713 |
| 0,38 | 0,722 | 0,712 |
| 0,40 | 0,700 | 0,702 |
| 0,42 | 0,717 | 0,683 |
| 0,44 | 0,705 | 0,674 |
| 0,46 | 0,757 | 0,694 |
| 0,48 | 0,703 | 0,708 |
| 0,50 | 0,670 | 0,695 |
| 0,52 | 0,677 | 0,700 |
| 0,54 | 0,682 | 0,661 |
| 0,56 | 0,732 | 0,689 |
| 0,58 | 0,705 | 0,677 |
| 0,60 | 0,671 | 0,717 |
| 0,62 | 0,701 | 0,651 |
| 0,64 | 0,673 | 0,697 |
| 0,66 | 0,681 | 0,695 |
| 0,68 | 0,670 | 0,683 |
| 0,70 | 0,651 | 0,660 |
| 0,72 | 0,680 | 0,679 |
| 0,74 | 0,674 | 0,648 |
| 0,76 | 0,680 | 0,696 |
| 0,78 | 0,663 | 0,680 |
| 0,80 | 0,661 | 0,606 |
| 0,82 | 0,649 | 0,663 |
| 0,84 | 0,652 | 0,624 |
| 0,86 | 0,618 | 0,659 |
| 0,88 | 0,651 | 0,620 |
| 0,90 | 0,667 | 0,642 |
| 0,92 | 0,665 | 0,649 |
| 0,94 | 0,628 | 0,642 |
| 0,96 | 0,639 | 0,618 |
| 0,98 | 0,616 | 0,613 |
| 1,00 | 0,607 | 0,611 |

Tabela C.2: Povprečen rang vrednotenja skupinskega filtriranja ob spremembah deleža upoštevanih podobnih študentov - vsakemu študentu je določenih po več testnih predmetov, iščemo podobnosti med študenti.

| **delež študentov** | **Jaccard** | **binary cosine** |
|---|---|---|
| 0,02 | 0,768 | 0,768 |
| 0,04 | 0,733 | 0,731 |
| 0,06 | 0,708 | 0,711 |
| 0,08 | 0,690 | 0,703 |
| 0,10 | 0,672 | 0,690 |
| 0,12 | 0,685 | 0,688 |
| 0,14 | 0,670 | 0,667 |
| 0,16 | 0,671 | 0,668 |
| 0,18 | 0,663 | 0,672 |
| 0,20 | 0,668 | 0,657 |
| 0,22 | 0,665 | 0,658 |
| 0,24 | 0,641 | 0,661 |
| 0,26 | 0,656 | 0,648 |
| 0,28 | 0,645 | 0,653 |
| 0,30 | 0,640 | 0,648 |
| 0,32 | 0,642 | 0,649 |
| 0,34 | 0,644 | 0,641 |
| 0,36 | 0,650 | 0,635 |
| 0,38 | 0,636 | 0,637 |
| 0,40 | 0,641 | 0,632 |
| 0,42 | 0,630 | 0,635 |
| 0,44 | 0,622 | 0,634 |
| 0,46 | 0,631 | 0,622 |
| 0,48 | 0,637 | 0,638 |
| 0,50 | 0,635 | 0,637 |
| 0,52 | 0,628 | 0,632 |
| 0,54 | 0,620 | 0,635 |
| 0,56 | 0,634 | 0,625 |
| 0,58 | 0,629 | 0,621 |
| 0,60 | 0,617 | 0,617 |
| 0,62 | 0,610 | 0,628 |
| 0,64 | 0,615 | 0,621 |
| 0,66 | 0,615 | 0,617 |
| 0,68 | 0,614 | 0,619 |
| 0,70 | 0,613 | 0,620 |
| 0,72 | 0,613 | 0,604 |
| 0,74 | 0,601 | 0,615 |
| 0,76 | 0,612 | 0,612 |
| 0,78 | 0,600 | 0,604 |
| 0,80 | 0,609 | 0,607 |
| 0,82 | 0,593 | 0,603 |
| 0,84 | 0,602 | 0,603 |
| 0,86 | 0,592 | 0,593 |
| 0,88 | 0,596 | 0,609 |
| 0,90 | 0,586 | 0,599 |
| 0,92 | 0,583 | 0,592 |
| 0,94 | 0,600 | 0,587 |
| 0,96 | 0,584 | 0,594 |
| 0,98 | 0,582 | 0,585 |
| 1,00 | 0,584 | 0,583 |

Tabela C.3: Povprečen rang vrednotenja skupinskega filtriranja - vsakemu študentu je določen po en testni predmet, iščemo podobnosti med predmeti.

| **številka testiranja** | **Jaccard** | **binary cosine** |
|---|---|---|
| 1 | 0,685 | 0,668 |
| 2 | 0,647 | 0,686 |
| 3 | 0,653 | 0,660 |
| 4 | 0,661 | 0,673 |
| 5 | 0,671 | 0,676 |
| 6 | 0,672 | 0,699 |
| 7 | 0,670 | 0,651 |
| 8 | 0,683 | 0,685 |
| 9 | 0,676 | 0,647 |
| 10 | 0,694 | 0,664 |

Tabela C.4: Povprečen rang vrednotenja skupinskega filtriranja - vsakemu študentu je določenih po več testnih predmetov, iščemo podobnosti med predmeti.

| **številka testiranja** | **Jaccard** | **binary cosine** |
|---|---|---|
| 1 | 0,614 | 0,618 |
| 2 | 0,605 | 0,613 |
| 3 | 0,604 | 0,614 |
| 4 | 0,605 | 0,616 |
| 5 | 0,614 | 0,622 |
| 6 | 0,606 | 0,608 |
| 7 | 0,604 | 0,617 |
| 8 | 0,604 | 0,614 |
| 9 | 0,612 | 0,610 |
| 10 | 0,618 | 0,611 |

# Dodatek D

# Izvorna koda algoritmov izračuna podobnosti

```
#racunanje podobnosti med predmeti s text-mining-om

import constructCorpus;
import math;

#seznam predmetov
txt="c:\\recommender_system\\subjects.txt"

#funkcija vrne podobnost dveh predmetov
#izracnano z dice-ovin koeficientom podobnosti
# 2 * presek mnozic / sum moci mnozic
def calculateDice(subject1, subject2, percent, corpus):

    set1=getKeywordsSet(subject1, percent, corpus)
    set2=getKeywordsSet(subject2, percent, corpus)

    diceCoef= float(2 * len(set(set1).intersection(set(set2)) ))
                    / float((len(set1) + len(set2)))

    return float(diceCoef)
#end calculateDice

#funkcija vrne podobnost dveh predmetov
#izracnano z Jaccard-ovim indexom podobnosti
# presek mnozic / unija mnozic
def calculateJaccard(subject1,subject2, percent, corpus):

    set1=getKeywordsSet(subject1, percent, corpus)
    set2=getKeywordsSet(subject2, percent, corpus)
```

```
    jaccardIndex= float(len(set(set1).intersection(set(set2)) ))
                        / float(len(set(set1).union(set(set2)) ))

    return float(jaccardIndex)
#end calculateJaccard

# funkcija vrne podobnost dveh predmetov
#zracnano s kosinusno podobnostjo - klasicna razlicica
def calculateCosineSimilarity(subject1,subject2,percent,corpus):

    #pridobimo slovarja (bag of words) z utezmi pri vsaki besedi
    set1=getKeywordsDict(subject1, percent, corpus)
    set2=getKeywordsDict(subject2, percent, corpus)

    commonSet=[];
    commonSet=set(set1.keys()).intersection(set(set2.keys()))

    sumWeights=0.0
    sumSub1=0.0
    sumSub2=0.0
    sum=0.0

    if(len(commonSet) != 0):
        for word in commonSet:
            sumWeights=sumWeights+
            (float(set1[word])*float(set2[word]))

            sumSub1=sumSub1+float(set1[word])*float(set1[word])
            sumSub2=sumSub2+float(set2[word])*float(set2[word])

        cosine = float(sumWeights)
                / float(math.sqrt(sumSub1)*math.sqrt(sumSub2))
    else:
        cosine=0.0

    return cosine
#end calculateCosineSimilarity

#funkcija vrne podobnost dveh predmetov
#-> izracnano s kosinusno podobnostjo - poenostavljena verzija
def calculateCosineSimilarity_2(subject1,subject2,percent,corpus):

    #pridobimo slovarja (bag of words) z utezmi pri vsaki besedi
    set1=getKeywordsDict(subject1, percent, corpus)
    set2=getKeywordsDict(subject2, percent, corpus)
```

```
    commonSet=[];
    commonSet=set(set1.keys()).intersection(set(set2.keys()))
    sumWeights=0.0
    sumSub1=0.0
    sumSub2=0.0
    sum=0.0

    if(len(commonSet) != 0):
        for word in commonSet:
            sum=sum+set1[word]*set2[word]

    cosine=round(sum,3)
    return cosine
#end calculateCosineSimilarity_2
```

calculateSimilarity_TM.py

```
#funkcija vrne podbnost med dvema studentom/predmetoma
#izracunano z Jaccardovim index-om (presek mnozic/unija mnozic)
#id = identifikator studenta/predmeta
#dict = podatki, ki jih obdelujemo (studenti ali predmeti)
#ignoreList = objekti, ki ga moramo ignorirati (testni predmet za
#katerega ugotavljamo, ali bi ga student vzel poleg ostalih ali ne)

def getJaccardIndex(id1, id2, dict, ignoreList):

    set1=dict[id1]
    set2=dict[id2]

    #print dict[id1]

    #testni predmet moramo izlociti:
    for ign in ignoreList:
        if(ign in set1):
            set1.remove(ign)

    jaccardIndex= float(len(set(set1).intersection(set(set2))))
                / float(len(set(set1).union(set(set2))))

    for ign in ignoreList:
        if(ign not in set1):
            set1.append(ign)

    return round(float(jaccardIndex),5)

#end getJaccardIndex
```

```
#Amazon's item-to-item algorithm
#funkcija vrne kosinusno podobnost izracunana na binarnih podatkih
#prirejeno funkcija zaradi drugane oblike podanih podatkov
#(nimamo 0/1 ampak seznam izbirnih predmetov)
#vhodni parametri -> glej getJaccardIndex
def getCosineSimilarity_binary(id1,id2,dict,ignoreList):

    set1=dict[id1]
    set2=dict[id2]

    #testni predmet moramo izlociti
    for ign in ignoreList:
        if(ign in set1):
            set1.remove(ign)

    intersaction = float(len(set(set1).intersection(set(set2))))
    product =

    float(math.sqrt(len(set1)))*float(math.sqrt(len(set2)))

    cosin = float(intersaction / product)

        #popravimo stanje
    for ign in ignoreList:
        if(ign not in set1):
            set1.append(ign)

    return round(float(cosine),5)

#end getCosineSimilarity_binary
```

calculateSimilarity_CF.py

# Literatura


[1] R. Farzan, P. Brusilovsky, "Social Navigation Support in a Course Recommendation System," v zborniku *Proceedings of the 4th international conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Springer-Verlag Berlin, Heidelberg, 2006, str. 91-100.

[2] H. F. Unelsrod, Design and Evaluation of a Recommender System for Course Selection. Trondheim - Norveška: Institutt for datateknikk og informasjonsvitenskap, 2011.

[3] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, *Recommender systems: An Introduction*. Cambridge [etc.]: Cambridge University Press, 2011.

[4] S. Džeroski, T. Erjavec, "Strojno učenje lematizacije neznanih slovenskih besed," v zborniku *Jezikovne tehnologije : zbornik konference*, Institut Jožef Stefan, Ljubljana, 2000, str. 14-30.

[5] C. D. Manning, P. Raghavan, H. Schütze. (1. april 2009). Introduction to Information Retrieval [Online]. Dostopno na: http://nlp.stanford.edu/IR-book/.

[6] P. B. Kantor, F. Ricci, L. Rokach, B. Shapira, "Content-based Recommender Systems: State of the Art and Trends," v *Recommender Systems Handbook*, P. B. Kantor, F. Ricci, L. Rokach, B. Shapira (ured.). London: Springer, 2011, str. 81-82.

[7] T. Segaran, "Making Recommendations," v *Programming Collective Intelligence*, M. Treseler O'Brien (ured.). Sebastopol: O'Reilly Media, 2007, str. 27-29.

[8] G. Linden, B.Smith, J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *Internet Computing, IEEE*, vol. 7, no. 1, str. 76-80, 2003.